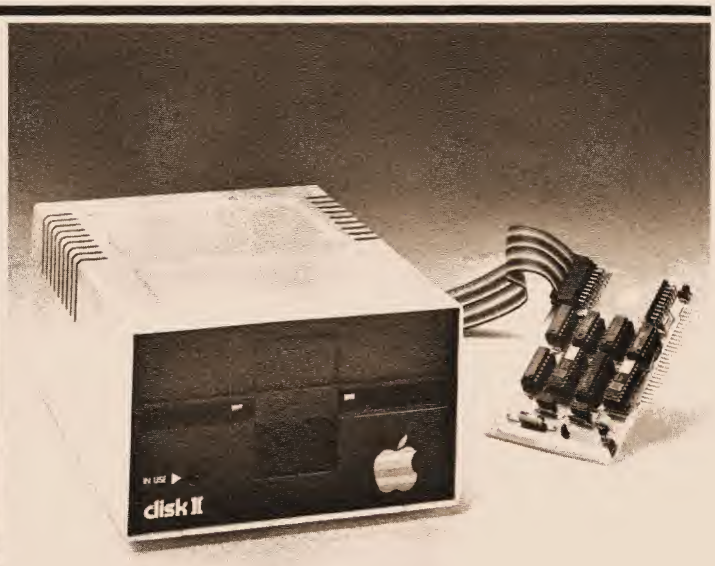
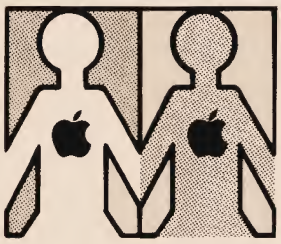
THE BEST OF
contact '78

the user group newsletters

apple computer inc.®





LOCAL USER GROUPS

Apple's still springing up

Many APPLE user groups have sprung up during 1978, and more are in the germination stage. Following is a list of those we knew about at the end of the year. If there is no group in your area and you want to start one, talk to your APPLE dealer. Chances are he knows most of the local APPLE II owners, and can help bring them together for your kickoff meeting. He might even want to host the meetings on a continuing basis. Ask him.

If you have formed (or want to form) a local user group, let us know. We can publicize your efforts and help you get off to a good start.

ALASKA

1. APPLE-HOLICS
   G. K. Inman
   SRA Box 1313
   Anchorage
   AK.
   99502
   (907) 344-1300

ALABAMA

2. APPLE CORPS
   Terry Woodward
   Computer Center, Inc.
   433 Valley Avenue Plaza
   Birmingham
   AL.
   35209
   (205) 942-8567

ARKANSAS

3. DATABITS
   C. Johnson
   C/O DATACOPE
   5706A W. 12th Street
   Little Rock
   AR.
   72204
   (501) 666-8588

CALIFORNIA

4. L. A. APPLE USERS GROUP
   11911 Wilshire Blvd.
   CA.
   90025

5. ABACUS USERS GROUP
   BYTE SHOP
   1122 B St.
   Hayward
   CA.
   94541
   (415) 886-2980

6. THE APPLE PICKERS
   SANTA ROSA COMPUTER CENTER
   604 Seventh Street
   Santa Rosa
   CA.
   95404

7. APPLE USERS GROUP
   SCOTT STARKWEATHER
   11074 San Pablo Avenue
   El Cerrito
   CA.
   94530
   (415) 233-5010

8. APPLE BYTE USERS GROUP
   LOY SPURLOCK
   APPLE BYTE PROGRAMMERS COMPUTER FORUM
   14052 E. Firestone Blvd.
   Santa Fe Springs
   CA.
   90670
   (213) 921-2111 & (714) 739-0711

9. APPLE CORE
   Marion A. Clarke
   COMPUTERLAND OF THOUSAND OAKS
   El Cid Plaza - 171 E. Thousand Oaks Blvd.
   Thousand Oaks
   CA.
   91360
   (805) 495-3554

10. APPLE-BIZ
    Melvin Wong
    301 Balboa
    San Francisco
    CA.
    94118
    (415) 221-8500

11. APPLE CORE AVIDD ELECTRONICS
    2210 Bellflower Rd.
    Long Beach
    CA.
    90815
    (213) 598-0444

12. APPLE CORE
    Scott Kamins
    Box 4816
    San Francisco
    CA.
    94101

13. APPLE USERS GROUP
    Mark Wozniak
    COMPUTERS PLUS
    1324 S. Mary
    Sunnyvale
    CA.
    94087
    (408) 735-1199

14. COMPUTERLAND OF LOS ALTOS
    Sarkis Kouzoujian
    4546 El Camino Real
    Los Altos
    CA.
    94022
    (415) 941-8154

15. NORTH ORANGE COUNTY COMPUTER CLUB
    David Smith
    607 North Twilight
    Placentia
    CA.
    92670
    (714) 993-9939

16. SILICON APPLE PROGRAMMING SOCIETY
2485 Rossotto Dr.
San Jose
CA.
95130
(408) 354-6120

17. VIDEO GAMES & COMPUTERS
301 Balboa
San Francisco
CA.
94118
(415) 221-8500

CONNECTICUT

18. APPLE USER GROUP
Glen Brennan
COMPUTERLAND OF FAIRFIELD
2475 Blackbrook Turnpike
Fairfield
CT.
06430
(208) 374-2227

DELAWARE

19. COMPUTERLAND OF NEWARK
James H. Higgins
Astro Shopping Center—
Kirkwood Highway
Newark
DE.
19711
(302) 738-9656

FLORIDA

20. Victor Steeb
SOUTHERN MICROCOMPUTER CO.
5901E Northwest 151st St.
Miami Lakes
FL.
33160
(305) 821-7401

21. APPLE USER GROUP
Pat Fiorentino
2201 Ponce De Leon Blvd.
Coral Gables
FL.
33134

FRANCE

22. APPLE OEDIP
Schraen Dominique
8 - Place Ste-Opportune - 75001
Paris
FRANCE
508-46-21 - 508-47-71.

GEORGIA

23. APPLE USER GROUP
Preston Love
DATAMART INC.
3001 N. Fulton Dr.
Atlanta
GA.
30305
(404) 266-0336

HAWAII

24. APPLE USER GROUP
Dennis Nyhagen
7110 C Ohana-Nui Circle
Honolulu
HI.
96818

IOWA

25. Earl Keyser
22 Clover Lane
Mason City
IA.
50428

IDAHO

26. Larry Bugbee
2874 Ithaca
Boise
ID.
83705
(208) 362-9132 (Home) &
(208) 384-6100 (Work)

ILLINOIS

27. APPLE PIE
Jerry Feil
17318 S. Locust
Tinley Park
ILL.
60477
(312) 532-8244 (Home)

28. NORTHWEST SUBURBAN APPLE USERS GROUP
Ken Rose & Steve Uhl
C/O COMPUTERLAND OF ARLINGTON HTS.
Arlington Hts.
ILL.
60004
(312) 359-6723 & (312) 882-4567

INDIANA

29. INDY APPLE PICKERS
Doug McIntosh
C/O HOME COMPUTER CENTER
2115 E. 62 Street
Indianapolis
IN.
46220

30. Joe Torzewski
51625 Chestnut Rd.
Granger
IN.
46530

LOUISIANA

31. SOUTHEASTERN SOFTWARE
7270 Culpepper Dr.
New Orleans
LA.
70126

MASSACHUSETTS

32. APPLESEED
Donald M. Isaac
17 Saxon Rd.
Worcester
MA.
01602

MARYLAND

33. MARYLAND APPLE CORPS.
Kevin Parks
COMPUTERS ETC.
13A Allegheny Avenue
Towson
MD.
21264
(301) 296-0520

MINNESOTA

34. MINI'APP'LES
Dan Buchler
13516 Grand Avenue South
Burnsville
MN.
55337
(612) 890-5051

MISSOURI

35. APPLE JACKS
Creighton Calfee
P.O. Box 8452
St. Louis
MO.
63132

NEW JERSEY

36. APPLE GROUP–NEW JERSEY
Steve Toth
1411 Greenwood Dr.
Piscataway
N.J.
08854
(201) 968-7498

37. COMPUTER LAB OF NEW JERSEY
Dan Fischler
141 Route 46
Budd Lake
NJ.
07828
(201) 691-1984

NORTH CAROLINA

38. APPLE CORE COMPUTER CLUB
Alex Popper
3915 E. Independence Blvd.
Charlotte
NC.
28205
(704) 523-5107

NEBRASKA

39. APPLESAUCE OF LINCOLN/OMAHA
Russ Genzmer
C/O TEAM ELECTRONICS
2055 'O' Street
Lincoln
NE.
68510

NEW MEXICO

40. THE APPLE CORPS
Earl J. Nielsen
PERSONALIZED COMPUTER SERVICES
1803 Corte Del Ranchero
Alamogordo
NM.
88310
(505) 437-8447

NEW YORK

41. NYC USERS GROUP
Neil Shapiro
C/O COMPUTER/MART OF NEW YORK
118 Madison Avenue
New York
NY.
10016
(212) 686-7923

42. Charles Kollett
32 N. Brewster Lane
Bellport–LI.
NY.
11713
(516) 286-0198

OHIO

43. APPLE-SIDER
John Anderson
5707 Chesapeake Way
Fairfield
OH.
45014
(513) 829-1340

OKLAHOMA

44. APPLE II USERS GROUP
Jerry Henshaw - PRESIDENT
C/O THE TULSA COMPUTER SOCIETY
P.O. Box 1133
Tulsa
OK.
74101
(918) 836-7364

OREGON

45. APPLE PORTLAND PROGRAM LIBRARY EXCHANGE
Ken Hoggatt - PRESIDENT
9195 S.W. Elrose Court
Tigard
OR.
97223
(503) 639-5505 (Home) &
(503) 644-0161 - X6136 (Work)

PENNSYLVANIA

46. COMPUTERLAND OF HARRISBURG
4644 Carlisle Pike
Mechanicsburg
PA.
17055

47. APPLE USERS GROUP
Neil Lipson
PHILADELPHIA AREA COMPUTER SOCIETY
29 S. New Ardmore Avenue
Broomall
PA.
19008
(215) 825-3800 - X278 (Work) & (215) 356-6183 (Home)

TENNESSEE

48. APPLE PI
Richard C. Secrist
(FORMERLY APPLEACHIAN USERS GROUP)
RT. #12 - Cherokee Hills
Sevierville
TN.
37862

TEXAS

49. APPLE BARREL
R. V. Collins
12502 Bexley
Houston
TX.
77099

50. THE APPLE CORPS.
Bobbie Ferrell
Greenhill School

14255 Midway Rd. - Fulton Bldg.
Dallas
TX.
75240
(214) 661-1211 (Work) &
(214) 243-6347 (Home)

51. APPLE CORPS.
COMPUTERLAND OF AUSTIN
3300 Anderson Land
Austin
TX.
78757
(412) 452-5701

52. APPLE SEED
Bill Hyde
THE COMPUTER SHOP
6812 San Pedro
San Antonio
TX.
78216
(512) 828-0553

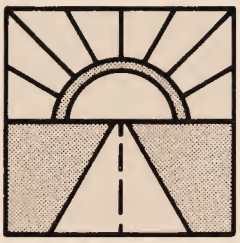
53. PHILIP W. JACKSON
C/O COMPUTER SOLUTIONS
Suite 124A - 9200 Broadway
San Antonio
TX.
78217
(512) 828-1455 & (800) 292-7652 (TOLL FREE)

WASHINGTON

54. APPLE PUGET SOUND PROGRAM LIBRARY EXCHANGE
Val Golding
6708 39th Ave. Southwest
Seattle
WASH.
98136
(206) 937-6588 (Home) &
(206) 623-7966

WISCONSIN

55. WISCONSIN APPLE USERS
Ken Blochowick
C/O Cybernetic Mechanism
P.O. Box 11463
Milwaukee
WI.
53211
(414) 964-6645



LOOKING AHEAD

...to how your Apple will grow

In case you are itchy to add to your Apple II system (or even just to learn more about it), here's what's coming up.

BASIC PROGRAMMING MANUAL—This is a beginner's guide to Apple BASIC, complete with lots of examples and illustrations. It will be mailed to all users who have warranty cards on file with us, starting late this month.

COMMUNICATIONS CARD—A card that lets your APPLE II talk over the phone with other computers will become available in April. For $180, this intelligent interface will allow the computer to control any 110 or 300 baud serial device through an industry standard RS-232C interface port. Look for a data-sheet in the next newsletter.

NEW APPLESOFT—It's coming, and it's going to be great! The new APPLESOFT fixes bugs in the existing version, and adds features such as data save/load, high-resolution plotting capability, ON ERR GOTO capability, and much more. It will be available on cassette in May, at $10. The ROM version, on a plug-in card (for slot #0), will be out in June at $99. The card will have a switch to allow you to choose between Apple BASIC and APPLESOFT.

FLOPPY DISK—The new mini-floppy should be on dealers' shelves by early July, at a price of less than $700 for the con-

troller card and one drive. (Each card will handle up to two drives.) The software, which works with either version of BASIC, will be able to load and store named files and provide disk directory lists. Look for more details two issues from now.

SERIAL INTERFACE CARD—This is a high-speed (to 9600 baud), programmable serial interface; designed to connect fast, half-duplex devices (printers, plotters, etc.) to the APPLE II. More details will be available two issues from now, and the device itself will be out in July. No price has been set.

EDITORIAL

by Phil Roybal,Marketing Mgr.

And now it's your turn

As we enter 1979, it is rewarding to look back and see how far we've come in these last twelve months. If you've joined our Apple family recently, I'd like to welcome you and share a little historical perspective with you.

At the start of 1977, personal computers sold largely to hobbyists. Most products were S-100 bus systems without the reliability, packaging, or software required to address other markets. Integrated systems (like APPLE II) were still a new idea. Indeed, some trade publications actually stated that such products were toys because they didn't have the expansion capability of the old S-100 systems! A year later, integrated systems dominate retail sales; and Apple has emerged as the premiere manufacturer in the quality-oriented segment of the market.

While that was happening, the market itself was changing character. Today there are more hobbyists than ever; but now they share the computer stores with businessmen, educators, and industrial users. This change has resulted in demands for useful application software that has put considerable pressure on our industry. While few programs actually emerged during 1978, a great number of them were specified and contracted for.

On the manufacturing side, the free ride was over. Poor product design, erratic quality, and inadequate financial strength took their toll. Some vendors fell prey to acquisition or reorganization, and ceased to be factors in the market. But the combined efforts of the industry sold around 180,000 systems. Apple Computer expanded production dramatically while building up an order backlog that reached 12 weeks during the Christmas buying spree. We grew rapidly; and profits were reinvested to finance production expansions, new product development, and better customer support.

The product line matured too. In January, we offered only the APPLE II system, at $1695 (16K RAM). By the end of the year we had introduced new manuals, modems, and printers; the Applesoft ROM Card; and our most popular peripheral, the floppy disk. Declining component costs and manufacturing economies of scale let us keep profit margins stable while we brought the price of a 16K system down to $1195. And most important, we were able to start several major software efforts that will yield dividends to every Apple user in '79.

What lies ahead? Throughout the industry, 1979 should be the Year of Software. More than two dozen companies presently market APPLE programs, and that number should swell to 100 during '79. We are planning several major announcements that span the gamut from languages and operating systems to business and education packages supplied ready-to-run. And those packages will be accompanied by documentation that sets the industry standard for completeness and clarity.

It's been a good year; and we owe a great deal to you, our customers. We plan to grow substantially during 1979 by bringing you products that will enhance the usefulness of your Apple investment. We thank you, and hope to continue earning your support.

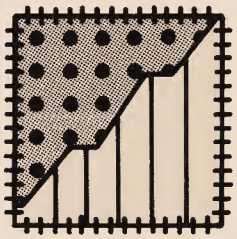
A FUNNY THING HAPPENED ON THE WAY HERE. . . .

This is the first of an irregularly published series of newsletters aimed at filling you in on what's happening at Apple; and at soliciting the feedback, product ideas, and people we need to service you, our customer base. In the past, we occasionally let our enthusiasm carry us away. It is often easy to forget how difficult it is to actually bring quality products into production. (And that's especially true when you're doubling in size every few months and still can't hire people fast enough to do everything that needs doing.) But on the way here, we've learned a bit. And we realize success hinges upon meeting commitments. So from now on, you'll see more conservative schedules from us; and we'll usually beat them.

This newsletter marks the start of an official APPLE II User Group: one that we trust will grow strong and healthy this year. But there's a lot of work yet to do. A major portion of it has to do with classifying and documenting all the software we have received over the past months. It really hasn't been lost. All 200 tapes are sitting in a box, waiting for us to look at them. While our goal was to publish a software list, description of the Software Bank, and Contributor's Form with this newsletter, we started too late. Instead, it'll be in the next newsletter. For sure.

Apple Computer has grown

from two men in a garage to 40 people in a modern 20,000 sq. ft. building, in less than two years. You made that possible, and we thank you. In the coming year we will continue to earn your support; and to actively solicit your inputs so that our efforts stay on target. It's important to us. We learned that on the way here.



PATCHES AND PROGRESS

...wherein program bugs are stomped upon

If (in the DOS) you try to load a program that had been saved under RAM APPLESOFT II, but you are now using the ROM card version, your program will not run correctly. To get around this problem, load your program and then type

CALL 54514

and your program will be correct.

Similarly, a program that was saved onto diskette from the ROM card version and later loaded under RAM APPLESOFT II will also cause problems. Simply type

CALL 3314

and your program will be correct. You can now save it onto tape.

The earliest production DOS had a problem: when in APPLESOFT II, any Read or Write statements with line numbers of 256 or higher would be ignored. To solve this problem on any disks you create from the DOS, bring up the DOS, remove the Write Protect sticker from your system's master diskette, and then type:

```
>PR#n (boot your system)
>BLOAD RAWDOS
>(hit Reset)
* 25D6:4C D5 3F
* 25DC:2E
* 3FD5:E8 F0 1 60 4C DD 25
* 3D0G
>BSAVE RAWDOS,
  A$1B00, L$2500
```

Any new masters created from this original master diskette will now work properly.

Cockpit errors and DOS

A high percentage – 75 per cent, to be precise – of disk errors found so far are due to users trying to run APPLESOFT II by typing "RUN APPLESOFT." This seems reasonable, but APPLE II doesn't see it quite that way. As a result (among other things) you cannot reload programs saved on disk.

To get things to come out right – all pointers where they should be, etc.–, from Integer BASIC simply type

FP (Carriage Return)

and you'll bring in APPLESOFT II, pointers and all.

To get back to Integer BASIC simply type INT.

Making life easier in APPLESOFT II

If you use APPLESOFT II and enter a statement such as IF X = A THEN PRINT T, APPLE won't do what you want it to do. That's because APPLESOFT II's parsing action causes the statement to be read as if you entered IF X = AT HEN PRINT T; a syntax error will appear and you won't know why.

To prevent this, simply enclose the A in parentheses, which stops APPLESOFT II's parsing of the letter combination. Thus, enter the statement as IF X = (A) THEN PRINT T, and all will be right with the world.

Had some printer card problems? Here's why, and the fix.

The original printer-card firmware uses the screen window width as the controlling parameter to set the margin for BASIC listings and TAB functions. This means that when your printer's line length is set at 132 columns, for example, the system display is set at 132 characters/line. We didn't think that this would cause a problem, because printer margins greater than 40 characters and screen displays are not allowed to co-exist.

But there's a catch when any screen clear functions are executed. The system uses window width to bound the clearing operations, which is a direct command not detectable by the printer card. But the keyboard input routine executes a "clear to end of line" when it gets a carriage return. Since the window width can be at well beyond the 40 columns, memory from the current cursor position to well past the normal boundary will be set to "space" ($A0). If the cursor is at the bottom of the screen, this can cause addresses from $800-up to be set to $A0. Simply setting the printer width on the keyboard with IC 132 N (CR), with the cursor at the bottom of the screen will cause these locations to be bombed because the window width gets set before the (CR) is executed. Since APPLESOFT starts at $800 and Integer BASIC variables start at $800, bad things will happen for sure.

So . . . to reduce the possibility of problems with APPLE's Parallel Printer card, do these things:

1–Home the cursor (ESC@, or CALL -936, in BASIC) prior to typing printer control sequences that set

the column width past 40 columns.

2—Do not use any screen or line clear operations when using the printer with the column width set past 40 columns. Then be sure to return the line length to 40 columns before turning off the printer card with PR#0.

If you're using the revised printer card with PROM P1-02, add this:

3—To perform the vertical tab in Integer or APPLESOFT BASIC on the printer, the command POKE 36, (Tab distance) should be used in place of the TAB or VTAB command.

HIRES demo tape type

Somehow we managed to ship a number of HIRES GRAPHICS demonstration tapes that carry a typographical error on the cassette labels. The error is in the brief loading instructions printed on the label: *8ØØ.FFFR. This should read: *CØØ.FFER. Note that the numeral "8" should be the letter "C". Programs on the tape will run correctly when you follow the proper loading instructions.

From time to time, we turn up a program bug. Although the fixes are incorporated into future versions of the program, often they are simple enough that users can "patch" their present programs to get improved performance. Here are patches for a popular program.

APPLESOFT CHANGES—The following patches to APPLESOFT will fix problems associated with the FRE, END, and DIM statements; and will allow the language to handle long program lines. The changes are made with a series of POKEs. They can be done from the command mode, or incorporated right into your program.

CHANGES
ARRAY INDEXING PROBLEM FIX:

POKE
6331,32:POKE6332,15Ø:POKE 6333,41:POKE6334,234

POKE
1Ø646,133:POKE1Ø647,177:POKE 10648,162:POKE1Ø649,5:POKE 1065Ø,165:POKE10651,132:POKE 10652,96

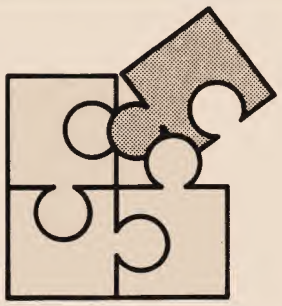
LONG LINE FIX:
POKE 3Ø5Ø,234
POKE 3Ø54, 136
POKE 3Ø55, 145
POKE 3Ø56, 158
POKE 3Ø57, 2Ø8
POKE 3Ø52, 251

'END' STATEMENT FIX:
POKE 2048,210

FRE( ) FUNCTION FIX:
POKE 6143,5

APPLESOFT MANUAL CORRECTION—A small typesetting disaster left us with an incomprehensible example on page 22 of the APPLESOFT Manual. The example should read as shown here.

```
100  DIM A$(15)
110  FOR I = 1 TO 15: READ A$(I):
      NEXT I
120  LET F = 0: I = 1
130  IF A$(I) <  = A$(I + 1) THEN
     180
140  LET T$ = A$(I + 1)
150  LET A$(I + 1) = A$(I)
160  LET A$(I) = T$
170  LET F = 1
180  LET I = I + 1: IF I < 15 GOTO
     130
190  IF F = 1 THEN 120
200  FOR I = 1 TO 15: PRINT A$(I)
     : NEXT I
220  DATA APPLE, DOG, CAT, RANDOM, CO
     MPUTER, BASIC
230  DATA MONDAY, "***ANSWER***", "
     FOO: "
240  DATA COMPUTER, FOO, ELP, MILWAU
     KEE, SEATTLE, ALBUQUERQUE
```



BITS AND PIECES

...things about Apple, handy to know

Document, document, document. It would be difficult for you to give us too much information about a program that you send up – we need to know as much (and sometimes more) about your program contribution as you can tell us. And never assume that potential users of your program are as clever as you; instead, assume that they know nothing – not even when to hit RETURN! Remember: The care you take in explaining your program is the only guarantee that it will be usable by anyone else.

SAVE, then SAVE again. SAVE your program twice, one recording after the other, on the same cassette. By doing this you give yourself, and us, a safety factor in that if our computer cannot read the first SAVE, perhaps it will be able to read the second. (As you know, not all recorders are created equal.)

By the way, please add your name and address to the program listing with REM statements before you save it, so that there will be no question as to whose program we're running when we try it out.

LOAD, check, then give yourself credit. After you SAVE your program, turn OFF the Apple to clear the memory space you've been using, then power it up and LOAD the program to be sure that it does in fact load and run.

Finally, protect. After saving your program, make sure that it cannot be erased. Every cassette includes a write/protect feature in the form of two small plastic tabs on the edge of the cassette opposite its business end. When these tabs are pushed in, pulled out, or otherwise gotten out of the way, the cassette can no longer be written into.

Clearing the air. We make no profit from your contributions to the user portion of the APPLE software bank. We set up this portion of the bank simply to encourage and to ease the exchange of programs among APPLE users. The honorarium that we pay for contributions to this portion of the bank is merely our way of encouraging such contributions.

From time to time, of course, APPLE does purchase software, and does so at the fair market value. Such purchases are negotiated individually, based upon market conditions, applicability of the product, etc.

The good earth

APPLE II ***must*** *be grounded —either via its three-wire power cord inserted into a properly grounded three-wire outlet, or by a wire inserted between APPLE's metal base plate and one of the plate's mounting screws. In the latter case, you must run the ground wire to a (preferably very close) ground — a long, long metal rod driven into the earth, and connected to APPLE via a heavy wire, or a wire connection to the cold-water system (but make sure you have metal piping). Grounding eliminates any possibility of the existence of a floating potential, which can be detrimental to you, to APPLE, and to proper tape cassette operation.*

Get your head straight

Cassette tape recorders — whether expensive or inexpensive — often suffer a misalignment of their playback head during shipping and other handling. Such misalignment causes azimuth error, which is death to the high-frequency response (particularly on units without a tone control) so necessary to accurately transfer data. But any audio shop technician can readjust the azimuth alignment; a skilled ear alone is often sufficient to do the job. Your original APPLE prerecorded tapes are excellent alignment references, because they are recorded with professional equipment maintained to the highest professional standards.

The colon as a listing formatter for Applesoft

The basic description of the colon's action is that it can be used to separate BASIC statements. But with the colon you can structure your listing in any way you desire, whether you need vertical spacing or horizontal spacing—or both, as in tabulating data into blocks of rows and columns.

To do it, enter the line number, then the colon. If there is no information following the colon on the line, then the display will step vertically. If information does follow the colon, then that information will be spaced to the right of the colon, allowing the interior code of FOR. . . NEXT loops, etc., to be neatly indented.

You can see how the colon is used for this purpose in this issue's *How To* section. Note the neat appearance of the listing.

Apple and Education

Apple Computer, Inc., announces the appointment of Roger Cutler as Education Specialist for the company. Roger is organizing an educators' user group. If you would like to be on his mailing list, please write him at Apple. Our plans include an educational software bank and advice on writing computer assisted instruction programs. Watch CONTACT for further information about educational applications of the Apple II.

CALLING ALL INTERFACE CARD DESIGNERS

Now that several other companies have begun to market interfaces for the APPLE II, a word on our design philosophy is in order.

We are very interested in any devices that plug into APPLE II, and are generally happy to provide information to prospective manufacturers. (Just write and ask for our Prototyping Board Write-up.) Our own philosophy is to build devices that are intelligent (have all control routines in on-board ROM) and are slot independent. However, we must sometimes compromise because of design considerations in the system. Therefore, we have made the following restrictions on slot assignments:

SLOT #	ASSIGNMENT
0	APPLESOFT BASIC ROM Card
5	Second Disk Controller Card
6	First Disk Controller Card

In addition, slots 4 and 5 are tentatively reserved for products that may have to be slot-dependent. Therefore, other manufacturers should design peripherals which are either completely independent of slot number, or will work in slots 1-3. In this way they will avoid possible conflicts with existing or proposed Apple Computer products.

OUT OF THE MIST....

...clearing up things thought already clear

Occasionally we document some wonderful feature in such a way that nobody understands it. As we identify those areas (recognizable by the stack of associated phone messages), this column will attempt to clarify them.

24K SYSTEM PROGRAM LOADING

Normally, entering BASIC with the B^{C} command resets the pointers to high and low memory (HIMEM & LOMEM), so that the monitor knows where to begin loading programs. However, 24K systems need a little human help to set the HIMEM pointer. Therefore, before loading or keying in programs, enter BASIC in the command mode

HIMEM:24576

When this step is omitted, the system believes it has a 32K memory to work in. Since BASIC programs are loaded starting from the top of memory, they tend to fall into the bit bucket and disappear forever. . . . .

PROGRAM LISTING WITH THE TTY PRINTER ROUTINE

The hardware/software TTY interface described on page 114 of the APPLE II REFERENCE MANUAL provides an inexpensive method of printing on a TTY (output only). The example on page 116 shows how commands can be incorporated into a BASIC program to produce hard copy output.

Unfortunately, we left out an example of how to LIST a program on the TTY. Here's how. Just enter the command mode of BASIC and type:

```
CALL 880
LIST
PR#0
```

That sequence will enable the TTY, list the program, and then return output to the TV screen.

USING THE HI-RES ROUTINES

A section of the new REFERENCE MANUAL describes the HI-RES plotting routines and mentions that they are available in ROM and on tape. Although the ROM's aren't available yet, many people don't know that they have the routines on tape already. These programs are the machine language load at the start of the HI-RES DEMO tape. If you load the machine language part and then skip loading the BASIC demo program, you will have the routines in memory to work with.

While use of most of the programs is straightforward, the SHAPE routine gives some people problems because they don't see how to build and use a shape table. Here's an explanation.

The SHAPE routine reproduces a figure from a set of instructions (the shape table) stored somewhere in memory. But it has to know where to find that table. It has been written to assume that the shape table begins at an address which is pointed to by memory locations 804 and 805. What you insert in those locations depends upon where you built your shape table.

Let's use the table example given in the REFERENCE MANUAL (page 53) to illustrate how to build an use the table. Here are the steps to follow:

1. Load the HI-RES routines into memory (COO.FFFR).
2. Build the shape table in memory, using the monitor. Let's arbitrarily start at address 900 hex, and fill in the data sequentially, just as shown in the manual: *900:12 3F 20 64 2D 15 36 1E 07 00 This will put the table above BASIC's variable space and below the bottom of a BASIC program.
3. Insert the shape table starting address into memory locations 804 and 805, as shown in line 10 of the sample program that follows (For more details, see the explanation in the *HOW TO* section under LOADING MACHINE LANGUAGE PROGRAMS. . . .). Then insert information on color, scale, rotation, etc., into the other memory locations specified in the REFERENCE MANUAL.

The following table presents decimal-number equivalents and APPLE keyboard equivalents to ASCII (American Standard Code for Information Interchange) characters. That is, if you were to scan the keyboard directly in BASIC, these are the characters that you would read; or, if you were to go into memory, these characters would appear as string-variable values.

In the APPLE keyboard column, read S^{M}C as a SHIFT/CONTROL-M B^{C} as CONTROL -B, etc. The decimal numbers listed are those before clearing of the keyboard strobe; after clearing of the strobe, the number is the listed value minus 128 (e.g., after clearing of the strobe, 145 becomes 17). If an APPLE keyboard equivalent doesn't exist, or if there is no decimal equivalent, then the table shows a double dash (--); if the keyboard equivalent is identical to the ASCII character itself, then the table shows "sa.".

Such information can make your programming life much easier, because it lets you get keyboard data directly into a

BASIC program without the use of an INPUT statement. As you know, INPUT statements can be limiting as, for example, when you type in a line and the screen yells SYNTAX ERROR at you. Well, what *is* the error? It may take a long time to find.

If, on the other hand, the keystrokes are picked directly off the keyboard then you – the programmer – are in command every step of the way; you make the decisions as to what goes and what does not, and in a way that tells you exactly *what's* wrong as you go wrong.

You can see one way in which this idea is applied in "Being Precise in INTEGER," in this issue's HOW TO section. In this Multiple Precision Arithmetic listing, statements 2500 2610 input data directly from the keyboard; they tell you exactly what's happening, the nature of any error you may have committed, etc. – and all by making use of the equivalence in the table presented here.

TABLE OF ASCII CHARACTER VALUES FOR INTEGER BASIC
(FOR APPLESOFT VALUES, SEE APPLESOFT MANUAL, APPENDIX K)

ASCII CHAR	APPLE KYBD	DEC EQUIV	ASCII CHAR	APPLE KYBD	DEC EQUIV	ASCII CHAR	APPLE KYBD	DEC EQUIV	ASCII CHAR	APPLE KYBD	DEC EQUIV
NUL	$_{P}C$	128	SP	Space Bar	160	@	sa.	192		--	224
SOH	A^C	129	!	sa.	161	A		193	a		225
STX	B^C	130	"		162	B		194	b		226
ETX	C^C	131	#		163	C		195	c		227
EOT	D^C	132	$		164	D		196	d		228
ENQ	E^C	133	%		165	E		197	e		229
ACK	F^C	134	&		166	F		198	f		230
BEL	G^C	135	'		167	G		199	g		231
BS	←	136	(		168	H		200	h		232
HT	I^C	137	)		169	I		201	i		233
LF	J^C	138	*		170	J		202	j		234
VT	K^C	139	+		171	K		203	k		235
FF	L^C	140	,		172	L		204	l		236
CR	M^C	141	-		173	M		205	m		237
SO	N^C	142	.		174	N		206	n		238
SI	O^C	143	/		175	O		207	o		239
DLE	P^C	144	0		176	P		208	p		240
DC1	Q^C	145	1		177	Q		209	q		241
DC2	R^C	146	2		178	R		210	r		242
DC3	S^C	147	3		179	S		211	s		243
DC4	T^C	148	4		180	T		212	t		244
NAK	U^C	149	5		181	U		213	u		245
SYN	V^C	150	6		182	V		214	v		246
ETB	W^C	151	7		183	W		215	w		247
CAN	X^C	152	8		184	X		216	x		248
EM	Y^C	153	9		185	Y		217	y		249
SUB	Z^C	154	:		186	Z		218	z		250
ESC	sa.	155	;		187	[	--	219			251
FS	--	156	<		188	\	--	220			252
GS	S_MC	157	=		189	]	S_M	221	ALT	--	253
RS	S_NC	158	>		190	↑	--	222		--	254
US	--	159	?		191		--	223	DEL RUBOUT	--	255

LF = Line Feed; CR = Carriage Return; SP = SPace; ESC = ESCape; sa. = Keyboard character same as ASCII character.

HOW TO ...things your manual never told you

LISTINGS—A LITTLE AT A TIME

There is a way to stop APPLE II's LIST operation. You must go into the MONITOR and enter the following bytes of hex:

```
3DA:A9 E3 85 36 A9 03 85
    37 60 RETURN
:48 AD 01 C0 10 08 AD 11
    C0 AD RETURN
:01 C0 10 FB AD 11 C0 68
    4C F0 RETURN
:FD 4C DA 03 RETURN
```

After loading the hex code, press CONTROL Y to activate this StopList routine.

Now we try listing memory by entering 0.FFF RETURN. Let some lines go by, then press any key. If you've done everything correctly, the scrolling will halt. Now press any other key and the lines will again scroll. This routine will still work when you go back into BASIC. But if for any reason you press RESET, you must reactivate the routine by going back into the MONITOR and pressing CONTROL Y.

How does it work? Well, to print a character on the screen, APPLE II uses a routine located in MONITOR. APPLE goes to the routine by finding its address in locations 54 and 55 (36 and 37 in hex). But entering a CONTROL Y replaces this address with the address of the Stop List routine. You catch APPLE because it always checks for a Stop List command before printing any character. Software engineers call such a feature a hook, and use it to make programming just a bit more handy.

Tony Hughes
THE APPLE CORE
San Francisco, Calif.

AN APPLESOFT CONVERT PROGRAM

Programs written in APPLESOFT BASIC and saved on tape cannot be LOADed and RUN with APPLESOFT II. But there is a way to use them without retyping—by means of our CONVERT program, which we list below.

The CONVERT program runs in Integer BASIC, accepts a tape in APPLESOFT BASIC, and produces a new tape in APPLESOFT II BASIC. To use CONVERT, LOAD the CONVERT tape. It will ask you if the old program (written in APPLESOFT BASIC) used OPTION 1 or OPTION 2.

- OPTION 1 was GRAPHICS COMMANDS WITHOUT LET OR REM STATEMENTS
- OPTION 2 was LET OR STATEMENTS, BUT NO GRAPHICS

After you answer you will be prompted to play the old program tape. After CONVERT has finished reading and processing the old tape, it will ask you to record a second tape. This second tape will be your original program converted into APPLESOFT II. If any errors are discovered during the conversion process, you will be given self-explanatory messages.

CONVERT

```
    0 TEXT : CALL -936: VTAB 3: PRINT "APPLESOFT CONVERSION PROGRAM:"
    2 PRINT "
CONVERTS OLD APPLESOFT PROGRAMS TO": PRINT "APPLESOFT II FORMAT"
    3 PRINT "
COPYRIGHT 1978 APPLE COMPUTER, INC.
"
    4 PRINT "

"  POKE 34,10
    5 PRINT "WAS PROGRAM WRITTEN IN OPTION 1 OR": PRINT "OPTION 2?
'  PRINT "OPTION 1: GRAPHICS COMM
     ANDS WITHOUT"
    6 PRINT "           LET OR REM STATEMENTS": PRINT "
OPTION 2: LET AND REM STATEMENTS BUT NO
          GRAPHICS "
    7 INPUT "OPTION #",O: IF O<>1 AND O<>2 THEN 7
   10 CALL -936: PRINT "PUT APPLESOFT PROGRAM TAPE IN RECORDER,": POKE 60,Z: POKE 61,Z: POKE 62
     ,2: POKE 63,Z:F=1536:B=4096
   20 INPUT "PRESS THE PLAY BUTTON, THEN HIT RETURN",A$: CALL -259
   25 IF PEEK (1)<128 THEN 30: PRINT "

TAPE READ ERROR!": PRINT "TRY RE-ADJUSTING VOLUME CONTROLS ON
      TAPEPLAYER, THEN RE-RUN THIS PROGRAM"
   30 POKE 60,Z: POKE 61,16:E= PEEK (Z)+ PEEK (1)*256-6657: POKE 62,E MOD 256: POKE 63,E/256:
     CALL -259
   35 CALL -936: PRINT "

CONVERTING..."
   40 IF B>=E THEN 1000:A= PEEK (B)+F MOD 256: POKE B,A MOD 256: POKE B+1, PEEK (B+1)+F/256+(A>
     255)
   50 FOR B=B+4 TO B+999:T= PEEK (B): IF T<133 THEN 250: IF T<>135 AND T<>142 OR O=2 THEN 200:
     C=B
   55 IF T<>142 THEN 60:T=137: GOTO 250
   60 C=C+1:U= PEEK (C): IF U=32 THEN 60: IF U=67 OR U=71 OR U=72 OR U=80 OR U=86 THEN GOTO U:
      PRINT "BAD STATEMENT IN PROGRAM": GOTO 250
   67 T=160: GOTO 90
   71 T=136: GOTO 90
   72 T=142: GOTO 87
   80 T=141: GOTO 90
   86 T=143
   87 CC=Z:D=B
   88 D=D+1: IF PEEK (D)<>44 AND PEEK (D)<>58 AND PEEK (D) THEN 88: IF PEEK (D)=44 THEN 89:
      PRINT "BAD STATEMENT IN PROGRAM!": GOTO 250
   89 CC=CC+1: IF CC=1 THEN 88: POKE D,197
   90 POKE C,32: GOTO 250
  199 REM : MAP OLD TOKENS TO NEW
  200 IF T>195 THEN 250:T=T+1+(T>134)*34+(T>139)+(T>160)+(T>177)*2
  250 POKE B,T: IF B/500*500=B THEN PRINT "STILL CONVERTING!"
  251 IF T<>0 THEN NEXT B:B=B+1: IF B<E THEN 40
  878 CC=Z:D=C
 1000 CALL -936: POKE 60,Z: POKE 61,Z  POKE 62,2: POKE 63,Z: PRINT "DONE!
"  INPUT "START RECORDING,
      THEN HIT 'RETURN'",A$
 1001 POKE E-2,Z  POKE E-1,Z: POKE E,Z
 1005 D=E-4096: POKE Z,D MOD 256: POKE 1,D/256: POKE 2,Z: CALL -307
 1010 POKE 60,Z: POKE 61,16: POKE 62,E MOD 256: POKE 63,E/256: CALL -307
 1020 PRINT "O.K.

   PRINT "THE TAPE JUST RECORDED CAN NOW BE LOADED INTO APPLESOFT II.": END
```

STRING + STRING = CONCATENATION

A string is a series of characters, zero to 255 characters in length. To symbolize that it is a string being operated upon, a string-variable name ends with the $ symbol. APPLE can operate on string variables, in whole or in part, just as it operates on numeric variables. And this ability to manipulate strings—to reformat them, etc.—is a powerful tool.

Concatenation is an important facet of string manipulations. To concatenate strings means to append one to another, to "series connect" hitherto independent character sets in order to operate upon them as a single entity. The concatenation function does exist in APPLESOFT and APPLESOFT II BASICs but, unfortunately, not in Integer BASIC.

There is, however, a simple routine that will let you concatenate strings with Integer BASIC. It's listed below, and operates by defining each string, finding the length of the first, and telling APPLE to tack the start of the second string to the end of the first string, and so on down the line. The example shown here is for two strings only; the maximum length of any concatenated string is, of course, limited to 255 characters.

```
 0 REM  EXAMPLE OF STRING CONCATENA
   TION IN INTEGER BASIC
10 DIM A$(40),B$(40)
20 A$="ABCDEF": REM  BUILD A$
30 B$="GHIJKL": REM  BUILD B$
40 PRINT "A$=";A$,"B$=";B$
50 L= LEN(A$): REM  FIND LENGTH OF
   CONTENTS OF A$
60 A$(L+1)=B$: REM  ADD B$ ON AFTER
    LAST CHAR IN A$
70 PRINT "A$=";A$,"B$=";B$: END
```

HOW TO GIVE A NUMBER SOME CHARACTER

Both APPLESOFT and APPLESOFT II contain the functions ASC and CHR$. These are opposites—complementary functions, if you will—in that ASC returns the decimal ASCII number equivalent of its designated string-variable argument, while CHR$ returns a single character equivalent of its designated decimal-ASCII number argument.

Examples: Say, M$="M" is some string variable of interest to us in APPLESOFT. When you tell APPLE to PRINT ASC (M$), APPLE responds with a "77" on its screen. (Remember, first character only; decimal 77 in ASCII corresponds to the letter "M".) On the other hand, tell APPLE to PRINT CHR$ (77), and you get the letter "M".

A rather simpleminded example of the use of this pair of functions could be to associate them with the RaNDom function in a program to generate random alphanumeric sequences, except that you cannot do it in Integer BASIC because, alas, Integer contains only the ASC function. But don't despair, for all is not lost. Below we give you a very short routine that gives Integer BASIC the equivalent of a CHR$ function. *Insert it in your programs wherever you wish to and—**presto!**—you can convert decimal numbers to their ASCII character equivalents.*

```
 0 REM  CREATEING A 'CHR$' FUNCTION
   , WHICH CONVERTS A NUMBER INTO I
   TS ASCII CHAR. EQUIVALENT
10 A$="": REM  MAKES A$ THE FIRST V
   ARIALBE DEFINED IN THE PROGRAM,
   SO WE KNOW WHERE IT IS
20 INPUT B: REM  GETS THE NUMBER TO
    BE CONVERTED. PRINTING CHARACTE
   RS ARE 161 AND UP
30 POKE 2053,B: REM  INSERTS NUMBER
    INTO THE STRING VARIABLE 'A$',
   SO IT CAN BE PRINTED AS A CHARAC
   TER
40 PRINT B;" CORRESPONDS TO '"
   ;A$;"'"
50 GOTO 20: END
```

HOW TO SET LOMEM WITHOUT HARDLY TRYING

LOMEM is the start of the Integer BASIC variables storehouse; HIMEM marks the top of the program store. Between the two is your working space. Entering BASIC with B^{C} will set LOMEM to 2048, the normal default value. But there are times (when using the Heuristics Speechlab™ for instance) when LOMEM must be set to a different value. The pro will incorporate such an operation right into his programs, rather than leaving it to chance.

Well, then, is there an easy way to reset LOMEM inside a program? The answer is yes, and we guarantee that your life will be simpler as a result. The example below will set a new LOMEM within an existing BASIC program. Of course, doing so will destroy existing variables, so do it before any are defined in the program. You can, in fact, insert this little routine at the start of your program so that it conveniently does the whole job for you.

1. Pick new LOMEM, l
2. a=l MOD 256: b=l/256
3. POKE 204,a: POKE 205,b—resets variable pointer
4. POKE 74,a: POKE 75,b—resets LOMEM pointer

EXAMPLE

1. New LOMEM: l-3000
2. a=184, b=11
3. POKE 204, 184: POKE 205, 11
4. POKE 74, 184: POKE 75, 11

Part of a personal computer's charm lies in the fact that it is a creative tool. Each person uses it

differently, to accomplish different goals. Unfortunately, this makes it difficult to write a manual that adequately covers everything a person might want to do with his system. The HOW TO section is therefore devoted to answering questions the manuals missed.

LOADING MACHINE LANGUAGE AS PART OF A BASIC PROGRAM

Often we want to include machine language data inside a BASIC program. A great many Apple tapes are made up this way to simplify the loading process. Here's a recipe for doing it yourself with programs written in Apple BASIC.

Apple BASIC loads programs into memory with the highest program line at the highest RAM location (HIMEM). Preceeding lines are located lower and lower in RAM. The beginning of the program is PP, an address which is held in memory locations CA and CB (hexadecimal), or 202 and 203, decimal. When you type SAVE, the computer transfers to tape everything between PP and HIMEM. Thus, to tuck machine language into your program so that it can later be loaded like BASIC, it is merely necessary to move the PP pointer down below the beginning of the extra code, put in two POKES to reset the pointer before running the program, and type SAVE. Later, you will be able to LOAD the whole thing just as if it were all BASIC. Just follow these steps:

1. Get the BASIC program into memory, just the way you want it. If you make any changes, you must re-do steps 2 and 6.
2. In the command mode, type:
   PRINT PEEK(202), PEEK (203)
   and write down the results. Let's call them m and n, respectively.
3. Load your machine language code into memory using the monitor load capability (xxxx.yyyyR). This will put the machine language program into memory below the beginning of the BASIC program, starting at hexidecimal address xxxx.
4. Take the starting address of the machine language program and divide it into two parts: xx xx. Convert each pair of digits from hex to decimal values: a & b; corresponding to the left and right xx pairs, respectively. Write them down.
5. Now enter the BASIC command mode and type:
   POKE 202, b-1 (value b from step 4, above)
   POKE 203, a (value a from step 4, above)
   POKE 204, 00
   POKE 205, 8
6. You have now moved the pointers down below your machine language program, and must insert code to move them back again when the program is run. To do that, type:
   0 POKE 202,m: POKE 203, n:GOTO q
   where m and n are the values from Step 2, and q is the first line number in your BASIC program. That line number can be 0—it will not be erased by the above entry.
7. Now you're done! Don't try to list your program before running it, because all you'll see is a meaningless set of numbers and symbols. Just type SAVE (before running the program), and it will all go onto tape. Later a LOAD command will bring it all back in.

CAUTION!

Once you have RUN such a program, you cannot SAVE it, for the pointers will have been moved. You can only save or copy a program like this before it has been RUN.

. . . A Moving Experience

On occasion it is handy to know how to move large blocks of data from one area of memory to another. While this is simple to do from the monitor (using the M command), it is a little more complex in BASIC. Below are some handles on the monitor routines that will let you use them within your BASIC programs.

When would you use them? Well, they come in handy for copying images from page 1 (normal) to page 2 in standard-resolution graphics. With two slightly-different images on the two pages, it is possible to do simple animations by switching rapidly back and forth between pages. (NOTE: Be sure to set LOMEM to 3072 or higher before using page two of graphics, or your variable table will be destroyed.) Here's how.

POKE 60, (old starting address mod 256)
POKE 61, (old starting address / 256)
POKE 62, (old ending address mod 256)
POKE 63, (old ending address / 256)
POKE 66, (new starting address mod 256)
POKE 67, (new starting address / 256)
CALL –468 (the actual move command)

Now, to use Page 2 (remember to set LOMEM to 3072 or higher):

```
10  POKE 60,0:POKE 61,4:
    POKE 62,255:POKE 63,7:
    POKE 66,0:POKE 67,8:
    CALL -468: POKE
    -16299,0
```

To switch back and forth between Pages 1 and 2:

POKE –16299,0 (displays Page 2)
POKE –16300,0 (displays Page 1)

If both pages contain similar graphics figures, then switching between the pages will yield simple animation; further effects may be gleaned from an inspection of the list of POKEs on page 30 of the APPLE II Reference Manual. *(NOTE: Don't try this with APPLESOFT in RAM. It starts at hex 800—the second page of graphic space. A block move into that area will send your APPLESOFT BASIC into the bit bucket!)*

PRINTING LOWER-CASE LETTERS WITH APPLE II

APPLE II cannot presently display lower-case characters on the screen, but it has no trouble printing them on most printers. To create a string of lower-case characters, simply generate the string in upper case at a known memory location; and then go through and add 32 to the ASCII value of each upper-case letter. That will produce the lower-case ASCII equivalent. If you then shove the modified code back into the string variable in place of the old value and print the string, you will print lower case characters.

As you convert the string, you must test each character to see that it is not a numeral punctuation, etc. Obviously, only alphabetic characters can be converted to lower-case. Here's a sample program that does the job.

This program works because, by defining A$ first, we know its absolute address in memory. Its first letter is stored in address 2053. (If this program were to be converted to APPLESOFT, we would have to manipulate our characters entirely through the string routines, since we cannot accurately locate the string in memory.)

```
 10 DIM A$(80): REM  BUFFER...MUST B
    E FIRST VARIABLE DEFINED IN PROG
    RAM
 20 INPUT "ENTER DATA: ";A$
 30 FOR I=1 TO LEN(A$)
 40 C= PEEK (2052+I): REM  SET C=ASC
    II VALUE OF THE "ITH" CHARACTER
    IN A$
 50 IF C<193 THEN 70: REM  TEST FOR
    A NON-ALPHA CHARACTER. IF FOUND,
     DO NOTHING
 60 POKE 2052+I,C+32: REM  CONVERT L
    ETTER TO LOWER CASE AND INSERT B
    ACK INTO A$
 70 NEXT I
 80 CALL -936: REM  CLEAR SCREEN
 90 PR#1: REM  TURN PRINTER ON (REME
    MBER, "PRINT D$;"PR#1" FOR DISK
    SYSTEMS
100 PRINT "80N": REM  SET PRINTER FO
    R 80 CHARACTERS
110 PRINT A$
120 PRINT "40N": REM  RESET PRINTER
    TO 40 CHARACTERS
130 PR#0: REM  SEE LINE 90 FOR DISK
    SYSTEMS
140 END
```

THE NAME OF THE GAME IS THE SAVING OF THE NAME

If you're an APPLESOFT II user working with, say, an inventory list with names, then you're in a bit of trouble if you want to SAVE the complete list to cassette tape, names and all: APPLESOFT II will save the numbers but not the names (strings). (Of course, the nicest way to SAVE such a list is to disk.)

If you need to save strings to tape however, the following program will do the job very nicely. Note that statement 10 creates space for the strings; 1010 gives you information about free memory space and how far up the variables are; 1050 writes out the length of the tape's string area; and 1070 writes all the desired information to tape.

Strings can also be created by establishing string pointers that point to the string definitions within the program itself rather than copying the string definition to high memory and pointing to it there. Strings created in this way are not saved by this technique and if the saved strings and pointers are used by a different program, these pointers will point to meaningless portions of the new program. This problem can be overcome by forcing the interpreter to copy the string item definitions to high memory. One way is to use concatenation:

```
20 A$(I) = A$(I) + " "
```

These two fixes should clear up all the problems in the program.

```
PR#0


]

?SYNTAX ERROR
]LIST

]  REM
2  REM PROGRAM TO SAVE STRINGS
3  REM  TO CASSETTE TAPE.
4  REM BY R. WIGGINTON (6/78)
5  REM
10  DIM A$(10)
20  PRINT "TYPE IN NINE STRINGS, SEPARATED BY": PRINT "CARRIAGE RETURNS. "
30  FOR K = 1 TO 9: INPUT A$(K): NEXT K
40  REM NOW SAVE A$ TO TAPE.
50  GOSUB 1000
55  PRINT "STRINGS ARE NOW ON TAPE.  TO RECALL,    TYPE 'GOTO 100', REWIND
      AND START TAPE, AND PRESS 'RTN'."
57  PRINT "LET TAPE RUN UNTIL CURSOR RETURNS."
60  END
100  REM THIS PART RECALLS THE
101  REM STRINGS FROM TAPE.
102  REM
110  DIM B$(10)
120  GOSUB 2000
130  FOR K = 1 TO 9: PRINT B$(K): NEXT K
140  END
1000  REM STORE A$ TO TAPE.
1003  PRINT "INSERT CLEAN TAPE, START RECORDING."
1005  PRINT "HIT ANY KEY WHEN READY": GET Z$
1010 X =  FRE (0): STORE A$: REM STORE A$ REALLY STORES POINTERS
1020  REM IN ORDER FOR THIS PROGRAM TO WORK, HIMEM MUST BE AT THE SAME
1021  REM VALUE WHEN THE STRINGS ARE RECALLED AS WHEN THEY ARE STORED.
1030 X =  PEEK (115) +  PEEK (116) * 256 -  PEEK (111) -  PEEK (112) * 256

1040  GOSUB 2100
1050  POKE 30,X -  INT (X / 256) * 256: POKE 31,X / 256: CALL  - 307: REM
     PUT (X) INTO LOCS 30&31, AND WROTE IT TO TAPE.
1060  REM (X) IS THE LENGTH OF THE STRING AREA.
1070  POKE 60, PEEK (111): POKE 61, PEEK (112): POKE 62, PEEK (115): POKE
     63, PEEK (116): CALL  - 307
1080  REM HAVE NOW WRITTEN EVERYTHING.
1090  PRINT "O.K.": RETURN
2000  RECALL B$: REM  GOT POINTERS BACK.
2010  GOSUB 2100: CALL  - 259: REM GOT LENGTH OF STRING AREA
2020 X =  PEEK (30) +  PEEK (31) * 256: REM X1SLENGTH OF AREA TO READ IN
2030 X =  PEEK (115) +  PEEK (116) * 256 - X
2040  POKE 60,X -  INT (X / 256) * 256: POKE 61,X / 256
2050  POKE 62, PEEK (115): POKE 63, PEEK (116): CALL  - 259
2060  RETURN
2100  POKE 60,30: POKE 61,0: POKE 62,31: POKE 63,0: RETURN : REM  SET CASS
     ETTE ROUTINE POINTERS.

]
```

Lining things up, point by point

Since most BASICs justify (i.e., line up) the left-most column, a display of multi-digit, decimal-pointed numbers can be awkward to read and somewhat unattractive.

A solution to this problem would be to use a tabulation routine that "justifies (or lines up) on the decimal point." Such a routine would position the numbers in a column so that the decimal points are vertically aligned. The short program listed below does exactly that.

Statements 10 through 50 are merely a demonstration routine that yields the sample run shown at the end of the listing. Statements 2000 through 2140 contain the routine that actually does the work. In effect, the routine aligns the numbers by right-justifying to the digit left of the decimal point, then tacks on the decimal point and the remaining digits to the right of the decimal point.

```
LIST
   10 F=-10: B=9: D=9
   15 A=F
   20 GOSUB 2000: PRINT ".";D
   30 IF A>3000 THEN 1000
   40 F=10*F: D=D+A
   50 GOTO 15
 1000 END
 2000 REM  RIGHT-JUSTIFICATION ROUTINE
      FOR APPLE BASIC
 2010 REM   INPUT IS ASSUMED TO BE IN
      VARIABLE "A"
 2020 REM  THE RIGHTMOST CHARACTER WILL
      APPEAR IN COLUMN CONTAINED IN
      VARIABLE "B"
 2100 A$=" ": IF A<0 THEN A$="-": REM
      GET SIGN OF NUMBER
 2110 A= ABS (A): REM  CONVERT A TO POS. #
 2120 C=(A>10)+(A>100)+(A>1000)+(A>10000):
      REM  DETERMINE HOW FAR TO
      LEFT-SHIFT PRINTOUT
 2130 TAB (B-C-1): PRINT A$;A;:
      REM  RIGHT-JUSTIFY PRINTOUT
 2140 RETURN
>RUN
       -10.9
      -100.19
     -1000.119
    -10000.1119
```

Being precise in INTEGER

The use of INTEGER BASIC limits you to the range of num-

bers between –32767 and +32767. Such a limitation is, at its best, frustrating, and, at its worst, infuriating. Consider, for example, the businessman who daily deals with foreign currencies, for which the basic monetary unit may be very, very small. What's a fella' to do?

Well, what he has to do is to go to multiple-precision arithmetic by means of a routine such as we present here. While this example is for addition only, it is readily adaptable to subtraction, multiplication, and division by changes in statements 3000 through 3080. The program does its job on large numbers in the same way as we do it by long-hand arithmetic; that is, it operates on one digit at a time, then carries to the next, and so on.

This particular listing is long and slow, because we wanted to make it clear and easy to read so that you could see what's happening. You may modify it to run much faster.

Incidentally, you can get a better understanding of the program's operation by relating certain of its statements to the ASCII conversion table that is in this issue's OUT OF THE MIST section. Statements 2500 through 2520, for example, result in the keyboard being read directly. 2540 refers to CHAR = 141; reference to the table tells you that decimal 141 is actually the Carriage Return. Similarly, 2545 excludes all characters except for the digits 0 through 9 (176 through 185). Again, statement 2550 converts the ASCII characters to the numbers themselves (i.e., if CHAR = 181, then 181 – 176 = 5).

```
** SYNTAX ERR
]LIST
   10 REM  MULTIPLE PRECISION ARITHMETIC
   20 REM  AN INTEGER BASIC EXAMPLE
   30 REM  THAT PROVIDES 20-DIGIT
   40 REM  ARITHMETIC PRECISION
   50 REM ------------------------------
  100 GOSUB 1000  REM  INITIALIZE EVERYTHING
  200 GOSUB GETA  REM  GET FIRST NUMBER INTO MATRIX A
  300 GOSUB GETB  REM  GET SECOND NUMBER INTO MATRIX B
  400 GOSUB ADDITION  REM  ADD MATRICES -- C=A+B
  500 GOSUB PUTC  REM  PRINT RESULT
  600 END
 1000 REM  INITIALIZATION ROUTINES
 1005 REM  ----------------------
 1010 DIM A(30),B(30),C(30),D(30),E(30)
 1020 GETA=2007 GETB=2100 PUTC=4010
 1030 ADDITION=3010
 1090 RETURN
 2000 REM  GET A ROUTINE
 2005 REM  ----------------------
 2007 PRINT "INPUT A",  TAB 20
 2010 GOSUB 2500  REM  GET INPUT INTO MATRIX E
 2020 FOR I=1 TO 30 A(I)=E(I)  NEXT I  REM  MOVE NUMBER INTO A
 2030 RETURN
 2100 PRINT   REM  GET B ROUTINE
 2103 REM  ----------------------
 2105 PRINT "INPUT B",  TAB 20
 2110 GOSUB 2500  REM  GET INPUT INTO MATRIX E
 2120 FOR I=1 TO 30 B(I)=E(I)  NEXT I  REM  MOVE NUMBER INTO B
 2130 RETURN
 2500 REM  KEYBOARD INPUT ROUTINE
 2505 REM  ------------------------
 2510 FOR I=1 TO 30 E(I)=0  NEXT I CHAR=0 DPTR=1  GOTO 2530
 2520 CHAR= PEEK (-16384)  REM  READ KEYSTROKE
 2530 POKE -16368,0  REM  CLR LAST KEYSTROKE
 2540 IF CHAR=141 THEN 2590  REM  GOT C/R
 2545 IF (CHAR<176 OR CHAR>185) THEN 2520  REM  NOT A NUMBER, SO IGNORE IT
 2550 D(DPTR)=CHAR-176  REM  CONVERT ASCII TO NUMBER AND SAVE
 2555 PRINT D(DPTR),  DPTR=DPTR+1
 2560 IF DPTR<21 THEN 2570  PRINT "INPUT TOO LONG--START OVER":  POP  POP
      END
 2570 CHAR=0  GOTO 2530  REM  WAIT FOR NEXT KEYSTROKE
 2590 EPTR=31 I=1  REM  WRAP UP  TRANSFER D INTO E
 2595 IF DPTR-I<0 THEN RETURN
 2597 E(EPTR-I)=D(DPTR-I)  I=I+1  GOTO 2595  REM  E=D, RIGHT-JUSTIFIED
 2610 RETURN  REM  LEAVE INPUT ROUTINES
 3000 REM  ADDITION ROUTINES
 3005 REM  ------------------
 3010 CARYIN=0
 3020 FOR I=30 TO 1 STEP -1
 3030 CARYOUT=0 TEMP=A(I)+B(I)+CARYIN  REM  ADD COLUMN PLUS CARYIN
 3040 IF TEMP<10 THEN 3060
 3050 CARYOUT=CARYOUT+1 TEMP=TEMP-10  GOTO 3040
 3060 C(I)=TEMP CARYIN=CARYOUT  REM  FINISHED ADJUSTING CARRY FIGURES
 3070 NEXT I
 3080 RETURN
 4000 REM  OUTPUT ROUTINE
 4005 REM  ------------------
 4010 PRINT  PRINT  PRINT
 4020 I=1
 4030 IF A(I)<>0 THEN 4040 I=I+1  GOTO 4030  REM  IGNORE LEADING ZEROS
 4040 TAB I+8  FOR J=I TO 30  PRINT A(J);  NEXT J
 4050 I=1
 4060 IF B(I)<>0 THEN 4070 I=I+1  GOTO 4060  REM  IGNORE LEADING ZEROS
 4070 PRINT  TAB I+8  FOR J=I TO 30  PRINT B(J);  NEXT J
 4080 I=1
 4090 IF C(I)<>0 THEN 4100 I=I+1  GOTO 4090  REM  IGNORE LEADING ZEROS
 4100 PRINT  TAB I+7  FOR J=I TO 31  PRINT "-";  NEXT J  PRINT
 4110 TAB I+8  FOR J=I TO 30  PRINT C(J);  NEXT J
 4115 PRINT  PRINT
 4120 PRINT  PRINT "****************************************":  PRINT
      RETURN

]RUN
INPUT A              9876543210
INPUT B              9999999999

                          9876543210
                          9999999999
                         ----------
                         19876543209

****************************************
                     1
```

The final Disk Operating System manual is in the works, and it's going to contain a lot of information. But like all good things, it will take some time. Meanwhile, we have produced some examples that show how to handle the most asked-about situations.

DATA FORMAT

The first thing to understand about the disk is its information format. Data can be written as fixed-length records or random-length records. All data is written in ASCII. Fixed-length records are written when you OPEN the file with an "L" parameter. OPEN DAN, L40 will create a file named DAN, whose records are all 40 bytes long. If you only put 20 bytes of information into each, you will waste 20 bytes per record of disk space.

Random-length records are actually one byte each, but are grouped together into blocks (logical records) which can be up to 32K bytes long. Each logical record ends with a carriage return. Note that this means the number "1" will require 2 bytes (number followed by a RETURN), and the number 10 will require 3 bytes. If you forget this and later replace "1" with "10", you will destroy part of the following record (poor programming practice).

USING RANDOM-LENGTH RECORDS

Example number one illustrates the writing of random-

length records. We simply open the file, then start printing. Each PRINT statement creates one logical record, ending with a carriage return. Thus, to put items into separate records, we must print them with separate PRINT statements. Each record is just long enough to hold the data you put in it.

Now run this program, using strings of less than 20 characters. It produces files called FILE and FILE.PTR, which are used in the following examples.

Example two reads the files created by example one. It first reads FILE.PTR, which tells it how long the data file will be. Then (in lines 90–100) it reads in that many records from FILE. This is a sequential read, where each INPUT statement brings in the next record.

Lines 210–220 are random access reads (this has nothing to to with random record length), where we read a particular record from the middle of the file. To do random reads, we specify the

```
5  REM  THIS PROGRAM SHOWS HOW TO
    WRITE RANDOM-LENGTH RECORDS
10  HOME : DIM A$(20),B$(20),
    A(20):N = 1
20  LET D$ = "": REM  CTRL D
30  PRINT "ENTER 'END' TO QUIT":
    PRINT "ENTER STRING #";N;:
    INPUT A$(N): IF
    A$(N) = "END" THEN 60
40  PRINT "ENTER ANOTHER STRING";:
    INPUT B$(N): PRINT "ENTER A
    NUMBER ";: INPUT A(N)
50  PRINT :N = N + 1: GOTO 30
60  PRINT D$"OPEN FILE"
70  PRINT D$"WRITE FILE"
80  FOR X = 1 TO N - 1
90  PRINT A$(X): PRINT B$(X):
    PRINT A(X): NEXT
100  PRINT D$"CLOSE  FILE"
110  PRINT D$"OPEN FILE.PTR"
120  PRINT D$"WRITE FILE.PTR"
130  PRINT N - 1
140  PRINT D$"CLOSE FILE.PTR"
150  END

]

5  REM  THIS PROGRAM SHOWS HOW TO
   READ BACK RANDOM-LENGTH RECORDS
10  LET D$ = "": REM  CTRL D
20  HOME : DIM C$(20),D$(20),C(20)
30  PRINT D$"OPEN FILE.PTR"
40  PRINT D$"READ FILE.PTR"
50  INPUT PTR
60  PRINT D$"CLOSE FILE.PTR"
70  PRINT D$"OPEN FILE"
80  PRINT D$"READ FILE"
90  FOR X = 1 TO PTR
100  INPUT C$(X): INPUT D$(X):
     INPUT C(X): NEXT
210  PRINT D$"READ FILE,R3"
220  INPUT C$(1),D$(1),C(1)
230  PRINT D$"CLOSE"
235  PRINT : PRINT : PRINT
240  PRINT C$(1),D$(1),C(1)
1000  END

]

]RUN RNDREADER
OPEN FILE.PTR
READ FILE.PTR
?3
CLOSE FILE.PTR
OPEN FILE
READ FILE
?THIS IS
?AN EXAMPLE
?1
?OF RANDOM
?RECORD
?2
?USAGE
?
?3
READ FILE,R3
?S IS
??AN EXAMPLE
??1
CLOSE


S IS  AN EXAMPLE     1
```

desired record, *minus 1*. Note however that the computer doesn't know how long each logical record is. (They're random length, remember?)

Therefore, it uses the *physical* record size, which is one byte long. (All records are physically one byte long, unless specified otherwise in an OPEN statement.) This means that if we say "READ FILE, R3," the next INPUT statement will start reading at the fourth *character* of the file. Try it and see.

USING FIXED-LENGTH RECORDS

Fixed-length records are all of the same size and format. They offer the advantage of ease of use, since they always present information the same way. On the other hand, they are inflexible. You must know when you start a file how big the largest record will be. Then, any smaller records will waste disk space (since all records are as long as the longest one). Example three shows the use of fixed-length records in both sequential and random access. Note that when writing into such a file (lines 120–150), we must specify each record number in a WRITE statement. Once the file is

```
10  REM   THIS PROGRAM PROVIDES AN
    EXAMPLE OF RANDOM RECORD ACCESS
40 :
60  REM  INITIALIZATION
70 : DIM A$(40):D$ =  CHR$ (4)
80 : PRINT D$;"NOMON C"
90  REM  CREATE FILE OF FIXED
    LENGTH, 30-BYTE RECORDS, EACH
    CONTAINING SIMILAR
100  REM  ASCII STRINGS.
110 : PRINT D$;"OPEN TEST2,L30"
120 : FOR I = 0 TO 5
130 : PRINT D$;"WRITE TEST2,R";I
140 : PRINT "NAME  ADDRESS ";I
150 : NEXT I
160  REM  NOW CHANGE ONE RECORD....
170 : PRINT D$;"WRITE TEST2,R3"
180 : PRINT "APPLE DOS  VER 3.1"
200  REM  NOW READ THE FILE,
     LOOKING FOR THE CHANGE.
     PRINT SOMETHING WHEN IT
210  REM  IS FOUND
220 : FOR J = 0 TO 5
230 : PRINT D$;"READ TEST2,R";J
240 : INPUT A$
250 : IF  LEFT$ (A$,5) = "APPLE"
     THEN  PRINT "THIS RECORD WAS
     CHANGED"
260 : NEXT J
270  REM  NOW CLOSE THE FILE, SO
     YOU WON'T GET AN 'OUT OF DATA'
     ERROR WHEN
      THE
280  REM  PROGRAM TERMINATES.
290 : PRINT D$;"CLOSE"
300  END


]RUN
NOMON C
NAME  ADDRESS 0
NAME  ADDRESS 1
NAME  ADDRESS 2
NAME  ADDRESS 3
NAME  ADDRESS 4
NAME  ADDRESS 5
APPLE DOS  VER 3.1
?NAME  ADDRESS 0
?NAME  ADDRESS 1
?NAME  ADDRESS 2
?APPLE DOS  VER 3.1
THIS RECORD WAS CHANGED
?NAME  ADDRESS 4
?NAME  ADDRESS 5
```

written, we can replace a random record with another that contains more characters (lines 160–180), so long as the new data does not exceed our 30-byte record length.

After a change is made, we can scan the whole file (lines 200–260) to find the change, read it, and do something about it.

EXECUTE FILES

The last example concerns itself with EXEC files. Unlike other files that contain programs or data, EXEC files contain *commands*, just *exactly* as they would by typed on the keyboard by a computer operator. Thus, commands in this file are not preceeded by a CTRL/D, since that's not what you would type on the keyboard. An EXEC file is created by a program made up for the purpose. It simply opens a file, and then PRINTs each command into it, just as you would have typed it. Example 4 shows a typical EXEC-builder program, which creates a file called COMMANDS. While most operations are straightforward, putting quotes into the file (for string printing, etc.) is tricky. Lines 70–90 show how to do it.

After this program has run, you will have an EXEC file on your disk. If you then say EXEC COMMANDS, it will take control of the system, do the specified operations, and return command to the keyboard when finished.

```
10  REM   THIS PROGRAM SHOWS HOW
      TO BUILD AN EXEC FILE
20  LET D$ =  CHR$ (4): REM CTRL
     D
30  PRINT D$;"OPEN COMMANDS"
40  PRINT D$;"WRITE COMMANDS"
50  PRINT "FP"
60  PRINT "FOR I=0 TO 10: PRINT I
     : NEXT I"
70  PRINT "RUN RNDREADER"
80  LET Z$ = "THIS IS THE END OF
     THE EXEC FILE."
90  LET Q$ =  CHR$ (34): REM HOW
     TO INSERT QUOTES
100  PRINT "PRINT";Q$;Z$;Q$
110  PRINT D$;"CLOSE"
120  END
```

Example 5 illustrates the use of the APPEND command. This command will open a file without "rewinding" it back to the beginning. Thus it allows you to build onto an existing file. The first item added to a file after it is appended will appear immediately following the last item of old data in the file. Try the following example to see how it works.

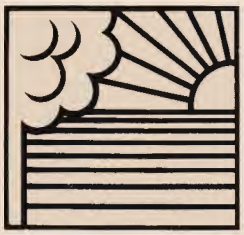
```
]LIST

100  REM   APPEND FILE

110  HOME : D$ = "": REM  CTRL D
120  INPUT "ENTER A STRING: ";A$
130  IF A$ = "" THEN 200
140  PRINT D$"APPEND TEST"
150  PRINT D$"WRITE TEST"
160  PRINT A$: PRINT D$"CLOSE"
170  GOTO 120
180 :
190 :
200  REM   GET IT BACK OUT

210  ONERR  GOTO 250
220  PRINT D$"OPEN TEST"
230  PRINT D$"READ TEST"
240  INPUT A$: GOTO 240
250  POKE 216,0: REM RESET FLAG
260  PRINT D$"CLOSE"
270 :
280 :
300  REM   THIS PROGRAM DEMON-
310  REM   STRATES THE USE OF
320  REM   THE 'APPEND' COMMAND
330 :
340  REM   IT WILL CONTINUE TO
350  REM   APPEND THE 'TEST'
360  REM   FILE UNTIL A NULL
370  REM   STRING IS ENTERED.
380 :
390  REM   AT THAT POINT IT WILL
400  REM   READ THE RECORDS BACK
410  REM   FROM THE FILE.

]
```



OUTSIDE THE ORCHARD

NEW PRODUCTS ... of interest to Apple users

This column is written as a service to Apple customers, and contains information on products that we feel to be of interest to the user community. Apple Computer Company does not in any way recommend these products, or warrant their suitability for use with the APPLE II computer.

NEW MODULATORS MEAN NO MORE WAVY LINES

Some APPLE II users have noticed wavy lines or color patches on their TV screens caused by radiation from airplanes, motors, or the computer itself. A UHF modulators seem to solve these problems neatly. One new device, called the Sup-R-Mod II, is available at $29.95, from

M&R ENTERPRISES
P.O. Box 1011
Sunnyvale, CA 94088

The new modulator comes completely assembled, ready to plug into the APPLE II.

Another modulator unit, this one from ATV Research, is available to transform APPLE II's video output to a video-modulated UHF signal. For B/W or color, the unit—called the Micro-Verter model MVX-500—outputs a signal tunable to one of four channels above television channel 14. It does not require direct connection to your television receiver's antenna terminals, and it is powered by four AA cells, which the manufacturer claims will last in excess of 1000 hours. Suggested price: $35, from dealers or factory direct. ATV Research, 13th and Broadway, Dakota City, NE 68731; (402) 987-3771.

HARDO-COPY GRAPHICS TERMINAL

The Panographic-84 is a precision x-y plotter with 100-step resolution in both directions; it is driven by zero-drift, adjustment-free stepping motors. Plotter programs can be written in BASIC or machine-language. Interface to APPLE is via our Parallel Printer card. Soon to be available is a chart reader that will allow you to use the plotter as an input device. The assembled plotter, with computer-operated pen lifter and molded cover, sells for $1400 (delivery 60 days ARO); as a kit, without lifter and cover ($995 (90-days delivery)); the pen lifter kit is $85; the cover, also $85. Pan Dynamics, Inc., 2950 Nebraska Ave., Santa Monica, CA 90404; (213) 829-2332.

TAPE RECORDER CONTROL

ROR and Candex Pacific have announced a relay activator to start and stop audio tape recorders via the REMOTE jack. The activator plugs into the GAME I/O connector on the Apple II, and is controlled with PEEK and POKE commands. Its connector allows the game controls to plug into it so that both devices can be used simultaneously.

Retail price is $24.95, and quantity discounts are available. Delivery is 30 days ARO. Technical questions should be addressed to Candex Pacific, 693 Veterans Blvd., Redwood City, CA 94063.

JOYSTICK UNIT

According to its manufacturer, this new full-range joystick has been designed to plug directly into APPLE's game I/O connector. With each joystick are two switches and two trimpots for adjusting to any APPLE II and any application. Completely assembled, the single joystick sells for $39.95; a double joystick is available at $79.95. Quantity discounts are available, and delivery is 30 days ARO. Microproducts Company, 1128 19th St., Santa Monica, CA 90403. (213) 393-8371.

Protect your investment from dust, coffee spills, and idle fingers with a heavy duty beige vinyl cover, custom fitted to your Apple II Computer. For only $6.95 (includes shipping) you can help to insure a long and useful life for your investment.

TO ORDER send check or money order to:

Henwood Enterprises, Inc.
1833 E. Crabtree Dr.
Arlington Heights, Ill. 60004

OR call TOLL FREE 800-323-7360 and use your Master Charge, VISA, or American Express credit card.

ANALOG INPUT CARD

The AI-02 is a single-card, 16-channel, analog data acquisition system for APPLE II. Each channel is individually addressable through software. The analog-to-digital conversion takes 70μs (8-bit resolution), after which an interrupt or a completion flag is activated and the converted value can be acquired by APPLE. Contact the manufacturer for more information on the AI-02, as well as for information on other of the firm's products for APPLE, such as a video input interface, and inventory and remote data entry software packages. Interactive Structures

Inc., Suite 204, Science Center, 3401 Market St., Philadelphia, PA 19104. (215) 382-8296.

INTERFACE BOARD

A newly announced peripheral interface board contains two 2716 PROMs, 20-mA and RS232 interfaces, a real-time clock, and a parallel output port. The board also has two 16-pin DIP sockets, which may be used to connect the board to an external source. Contact the manufacturer for pricing information on the complete board. A bare board—without components, but with assembly instructions, diagrams, and programs—is available for $29. Delivery, 3–4 weeks ARO. Peripheral Interface, 173–1128 McKercher Dr., Saskatoon, Sask. Canada S7H 4Y7.

THE GANG OUTSIDE THE ORCHARD

Here is a list of over 50 companies or groups that market software, hardware, or newsletters for APPLE II owners.

ARIZONA

1. PROGRAMMERS SOFTWARE EXCHANGE
   2110 North 2nd St.
   Cabot
   AR.
   72023

CALIFORNIA

2. AJA SOFTWARE
   P.O. Box 2528
   Orange
   CA.
   (714) 774-1270

3. APPLE CORE
   Scott Kamins
   P.O. Box 4816
   San Francisco
   CA.
   94101

4. ASTRO GRAPHICS SOFTWARE
   140 Willow Ave. #2
   Fairfax
   CA.
   94930

5. CANDEX PACIFIC
   693 Veterans Blvd.
   Redwood City
   CA.
   94063
   (415) 364-8427

6. CHARLES MANN & ASSOCIATES
   1926 South Veteran Ave.
   Los Angeles
   CA.
   90025
   (213) 473-0244

7. COMPUTER COMPONENTS INC.
   6791 Westminster Ave.
   Westminster
   CA.
   92683

8. DAVID GORDON
   David Gordon
   16956 Tupper St.
   Sepulveda
   CA.
   91343

9. ED AVELAR
   Ed Avelar
   2850 Jennifer Dr.
   Castro Valley
   CA.
   94546
   (415) 538-1235 (9-5)

10. G2 PROGRAM LIBRARY - GRT CORP.
    1286 North Lawrence Station Rd.
    Sunnyvale
    CA.
    94086

11. GEORGE W. LEE
    George W. Lee
    18803 S. Christina Ave.
    Cerritos
    CA.
    90701
    (213) 865-1639

12. HEURISTICS INC.
    900 North San Antonio Rd.
    Los Altos
    CA.

13. INNOVATIVE COMPUTER PROGRAMS
    Ron Graff
    P.O. Box 622
    El Toro
    CA.
    92630
    (714) 586-2246

14. M & R ENTERPRISES
    P.O. Box 1011
    Sunnyvale
    CA.
    94088

15. MICROPRODUCTS
    1024 - 17th St.
    Hermosa Beach
    CA.
    90254
    (213) 374-1673

16. MOUNTAIN HARDWARE INC.
    5523 Scotts Valley Dr.
    Scotts Valley
    CA.
    95066
    (408) 438-4734

17. PROGRAMMA CONSULTANTS
    3400 Wilshire Blvd.
    Los Angeles
    CA.
    90010

18. QUALITY SOFTWARE
    10051 Odessa Ave.
    Sepulveda
    CA.
    91343

19. RAINBOW COMPUTING INC.
    10723 White Oak Ave.
    Granada Hills
    CA.
    91344
    (213) 360-2171

20. SOFTAPE SOFTWARE EXCHANGE
    10756 Vanowen St.
    North Hollywood
    CA.
    91605
    (213) 985-5763

21. STRICTLY SOFTWARE
16720 Hawthorne Blvd.
Lawndale
CA.
90260
(213) 371-7144

22. WISE OWL WORKSHOP
Clifford T. Schafer
1168 Avenida De Las Palmas
Livervore
CA.
94550

CANADA

23. SPEAKEASY SOFTWARE LTD.
P.O. Box 1220
Kemptville - Ontario
Canada
KOG 130
(613) 258-2451

CONNECTICUT

24. PROGRAM DESIGN INC.
Carol Klitzner
11 Idar Court
Greenwich
CT.
06830

FLORIDA

25. DR. GEORGE L. HALLER
Dr. George L. Haller
1500 Galleon Dr.
Naples
FL.
33490

26. TRANS-DATA CORP.
161 Almeria Ave. - Dept. B-118
Coral Gables
FL.
33134
(305) 576-7666

ILLINOIS

27. ELECTRIC KEYBOARD LTD.
1920 N. Lincoln Ave.
Chicago
ILL.
60614
(312) 751-1555

28. HENWOOD ENTERPRISES INC.
1833 E. Crabtree Dr.
Arlington Hts.
ILL.
60004
(800) 323-7360

29. WALLACE ELECTRONICS INC.
4921 N. Sheridan Rd.
Peoria
ILL.
61614
(309) 692-2616

IOWA

30. EARL KEYSER
Earl Keyser
22 Clover Lane
Mason City
IA.
50401

LOUISIANA

31. SOUTHEASTERN SOFTWARE
7270 Culpepper Dr.
New Orleans
LA.
70126

MARYLAND

32. COMPUTERS ETC.
9330 Georgia Ave.
Silver Spring
MD.
20910
(310) 588-3748

33. MICRO USERS - SOFTWARE EXCHANGE INC.
7112 Darlington Dr.
Baltimore
MD.
21234
(301) 661-8531

34. U ASKED 4 IT
Stuart Frager
P.O. Box 13331
Baltimore
MD.
21203

MASSACHUSETTS

35. ERIC ROSENFELD
Eric Rosenfeld
70 Lancaster Rd.
Arlington
MA.
02174

36. PERSONAL SOFTWARE
P.O. Box 136
Cambridge
MA.
02138
(617) 783-0694

MICHIGAN

37. SYMETEC INC.
P.O. Box 462
Farmington
MI.
48024

MINNESOTA

38. NELSON R. CAPES
Nelson R. Capes
586 Kent Lane
Shoreview
MN.
55112

MISSOURI

39. MILLIKEN COMMUNICATIONS CORP.
Bodie Marx
1100 Research Blvd.
St. Louis
MO.
63132
(800) 325-4136

NEBRASKA

40. ATV RESEARCH
13th and Broadway
Dakota City
NE.
68731
(402) 987-3771

NEVADA

41. 6502 PROGRAM EXCHANGE
2920 Moana
Reno
NV.
89509

NEW JERSEY

42. CREATIVE COMPUTING SOFTWARE
P.O. Box 789-M
Morristown
NJ.
07960
(800) 631-8112 - in NJ.
(201) 540-0445

NEW YORK

43. EBC
P.O. Box 138
Freeville
NY.
13068

44. PRS—THE PROGRAM OF THE MONTH CORP.
257 Central Park West
New York
NY.
10024

PENNSYLVANIA

45. ARESCO - THE RAINBOW MAGAZINE
P.O. Box 43 - Dept.
RP-1178
Audubon
PA.
19407

46. INTERACTIVE STRUCTURES INC.
3401 Market St. - Suite 204
Science Center
Philadelphia
PA.
19104
(215) 381-8296

47. MICROTRONIX
P.O. Box Q - Dept. S
Philadelphia
PA.
19105
(800) 523-4550

TEXAS

48. B & G INTERFACE
P.O. Box 59364
Northhaven Station
Dallas
TX.
75229

49. DALLAS APPLE CORP. - THE GREENHILL SCHOOL
Bob Matzinger
P.O. Box 13446
Arlington
TX.
76013

50. S-C SOFTWARE
P.O. Box 5537
Richardson
TX.
75080

UTAH

51. BASIC BUSINESS SOFTWARE CO.
P.O. Box 2032
Salt Lake City
UTAH
84110

VIRGINIA

52. CH GALFO
602 Orange St.
Charlotteville
VA.
22901
(804) 296-4832

53. HOME COMPUTER CENTER INC.
2927 Virginia Beach Blvd.
Virginia Beach
VA.
23452
(804) 340-1977

WASHINGTON

54. APPLE PUGET SOUND PROGRAM LIBRARY EXCHANGE
Val Golding
6708 39th Ave. S.W.
Seattle
WA.
98136
(206) 937-6588 (Eves.)
(206) 623-7966 (Days)

55. DARRELL'S APPLEWARE HOUSE
17638 - 157th Ave. S.E.
Renton
WA.
98055

56. PUGETSOUND PROGRAM LIBRARY EXCHANGE
6708 - 39th Ave. S.W.
WA.
98136
(206) 932-6588

DIRECTORY

Here is the start of what we hope will become a comprehensive list of APPLE software available from outside sources. If you know about a product that should be on this list and isn't, please tell us about it. These are the ones we knew about in December, 1978.

EDUCATION

PROGRAM	MEMORY	MEDIA	PRICE	SOURCE
BASIC TUTORIAL		TAPE	$ 9.95	AJA SOFTWARE
ENGLISH GRAMMAR 1	IN PILOT	TAPE	$ 10.00	Earl Keyser
ENGLISH GRAMMAR 2	IN PILOT	TAPE	$ 10.00	Earl Keyser
ENGLISH GRAMMAR 3	IN PILOT	TAPE	$ 10.00	Earl Keyser
ENGLISH GRAMMAR 4	IN PILOT	TAPE	$ 10.00	Earl Keyser
ENGLISH GRAMMAR 5	IN PILOT	TAPE	$ 10.00	Earl Keyser
ENGLISH GRAMMAR 6	IN PILOT	TAPE	$ 10.00	Earl Keyser
ENGLISH GRAMMAR 7	IN PILOT	TAPE	$ 10.00	Earl Keyser
ENGLISH GRAMMAR 8	IN PILOT	TAPE	$ 10.00	Earl Keyser
ENGLISH GRAMMAR 9	IN PILOT	TAPE	$ 10.00	Earl Keyser
ENGLISH GRAMMAR 10	IN PILOT	TAPE	$ 10.00	Earl Keyser
GRADING ROUTINE	16k	TAPE	$ 7.95	Innovative Computer
GARDENING		TAPE	$ 10.00	Earl Keyser
IQ BUILDER: VOCABULARY		TAPE	$ 12.50	Program Design
IQ BUILDER: Number Series		TAPE	$ 9.50	Program Design
IQ BUILDER: Analogies		TAPE	$ 9.50	Program Design
INTEGER BASIC TUTORIAL		TAPE	$ 17.50	Puget Sound Program
LEARNING BASIC	16k	TAPE	$ 19.95	Innovative Computer
MORSE CODE TRAINER		TAPE	$ 10.00	Rainbow Computing
Matching Quiz	8k	TAPE	$ 6.95	Innovative Computer
Memory Aid	8k	TAPE	$ 6.95	Innovative Computer
Preschool IQ Builder		TAPE	$ 9.95	Program Design
Step-by-Step BASIC		TAPE	$ 29.95	Program Design
Story Builder/Word Master		TAPE	$ 9.50	Program Design
Study Aid	8k	TAPE	$ 6.95	Innovative Computer
Sell-3 Economic Simulators		TAPE	$ 10.00	Earl Keyser
Vocabulary Word Games (10)		TAPE	$ 15.00	Earl Keyser
Workshop Cassette		TAPE	$ 5.00	Puget Sound Program
Wordsmith, Wumpus		TAPE	$ 9.95	U ASKED 4 IT

FINANCIAL CALCULATIONS & DATA HANDLING

PROGRAM	MEMORY	MEDIA	PRICE	SOURCE
Apple Disk Business Systems		DISK	$ 35.00	AJA SOFTWARE
Apartment Building Cost Analysis (needs Applesoft)		TAPE	$ 15.00	Rainbow Computing
Asset Record Program			$ 49.95	Charles Mann
Apartment Billing		TAPE	$ 50.00	Darrell's Appleware
Amortization Schedule		TAPE	$ 24.00	Programmer Software
Business Inventory	10k	TAPE	$ 40.00	Computing Components
Business Inventory Package		TAPE	$160.00	Darrell's Appelware
Banking & Financial (20 programs)		TAPE	$ 19.95	Ed Avelar
Budget Plan		TAPE	$ 5.95	AJA Software
Billings Management			$ 89.95	Charles Mann
Check Verification Program			$ 49.95	Charles Mann
Check Book		TAPE	$ 20.00	Trans-data
Check Book		DISK	$ 28.00	Trans-data
Check Book		TAPE	$ 20.00	Rainbow Computing
Data Management Program			$ 24.98	Charles Mann
Expense Account Records		TAPE	$ 50.00	Darrell's Appleware
Executive Management Overview			$ 49.95	Charles Mann
Financial Tutorial		TAPE	$ 24.00	Speakeasy Software
Filing System Cross-Reference File		TAPE	$ 50.00	Darrell's Appleware
File Use Tutorial		TAPE	$ 15.00	Trans-data
File Use Tutorial		DISK	$ 23.00	Trans-data
Finances		TAPE	$ 15.00	Trans-data
Finances		DISK	$ 23.00	Trans-data
Home Improvement Records		TAPE	$ 50.00	Darrell's Appleware
Home Inventory Records		TAPE	$ 50.00	Darrell's Appleware
Home Financial Record Program		TAPE	$ 20.00	Computer Components
Inventory of Goods		TAPE	$ 20.00	Earl Keyser
Inventory			$ 89.95	Charles Mann
Invoicing Program			$ 59.95	Charles Mann
Income Tax-1040, A & B		TAPE	$ 25.00	Rainbow Computing
Inventory		TAPE	$ 35.00	Rainbow Computing
Inventory Package (Sort/POS/ Recorder)			$125.00	Darrell's Appleware
Ledger Record System			$ 59.95	Charles Mann
Label Print		TAPE	$ 10.00	Trans-data
Label Print		DISK	$ 18.00	Trans-data
Mailing List System		TAPE	$ 50.00	Trans-data
Mailing List Management			$ 79.95	Charles Mann
Professional Secretary Package			$ 89.95	Charles Mann
Personal Secretary Package			$ 59.95	Charles Mann
Professional Time Management			$ 59.95	Charles Mann
Program 200 Maintain Inventory File		TAPE	$ 50.00	Darrell's Appleware

FINANCIAL CALCULATIONS & DATA HANDLING

PROGRAM	MEMORY	MEDIA	PRICE	SOURCE
Program 205 Sort on Part Number		TAPE	$ 20.00	Darrell's Appleware
Program 210 Print Sales Slips & Updates		TAPE	$ 50.00	Darrell's Appleware
Program 220 Generates Recorders Report		TAPE	$ 50.00	Darrell's Appleware
Retail Management System			$ 59.95	Charles Mann
Software Pac #3 Statistics		DISK	$ 30.00	Basic Business
Software Pac #4 Finance Calculator		TAPE	$ 15.00	Basic Business
Soft II, Financial Pkg & Golf Handicap		TAPE	$ 10.00	Dr. George L. Haller
Stock Market Anlysis		TAPE	$ 9.95	AJA Software
Tax Planning Program			$ 89.95	Charles Mann
Tax Planning		TAPE	$ 7.95	Ed Avelar
Universal Database		TAPE	$ 60.00	Darrell's Appleware
Vendor File		TAPE	$ 50.00	Darrell's Appleware

SCIENTIFIC CALCULATION & MATHEMATICS

PROGRAM	MEMORY	MEDIA	PRICE	SOURCE
METRIC Conversion		TAPE	$ 20.00	Trans-data
METRIC Conversion		DISK	$ 28.00	Trans-data
Math Practice		TAPE	$ 5.00	Programmer Software
Number Converter Bases		TAPE	$ 5.00	Astro Graphics
Number Cruncher			$ 9.95	Micro Users
Super Math		TAPE	$ 18.00	Trans-data
Super Math		DISK	$ 25.00	Trans-data
Super Math	16k	TAPE	$ 7.95	Innovative Computer
Talking Calculator		TAPE	$ 12.95	Softape Software

LANGUAGES

PROGRAM	MEMORY	MEDIA	PRICE	SOURCE
Apple FORTH		TAPE	$ 37.00	Programma Consultants
Co-resident Assembler				Microproducts
FOCAL		TAPE	$ 25.00	The 6502 Program Exc
MICRO PRODUCTS Apple Assembler		TAPE	$ 20.00	Rainbow Computing
PILOT Interpreter		TAPE	$ 20.00	Earl Keyser
6k Assembler/Text Editor		TAPE	$ 29.95	ARESCO

ENTERTAINMENT

PROGRAM	MEMORY	MEDIA	PRICE	SOURCE
Appletalker		TAPE	$ 15.95	Softape Software
Appledian–Irish Jig		TAPE	$ 10.00	Rainbow Computing
Applevision–HIRES GRAPHIC/ MUSIC DEMO		TAPE	$ 15.00	Rainbow Computing
Astrology (Emphemer is needed)		TAPE	$ 10.00	ASTRO Graphics
Artillery Duel		TAPE	$ 10.00	Programmer Software
APPLEODIAN--5 OCTAVES & 12 VOICES		TAPE	$ 2.00	*SOFTAPE SOFTWARE
AWARI, TOWER OF HANOI, HEXPAWN		TAPE	$ 2.00	*SOFTAPE SOFTWARE
Bridge Challenger	16k	TAPE	$ 14.95	Personal Software
Bowling		TAPE	$ 5.95	AJA Software
Bingo		TAPE	$ 5.95	AJA Software
Blackjack		TAPE	$ 10.00	Rainbow Computing
Blackjack		TAPE	$ 2.00	*Softape Software
Bob Bishop's Hires Starwars		TAPE	$ 15.00	COMPUTER COMPONENTS
Bob Bishop's Hires Rocket Lander		TAPE	$ 15.00	COMPUTER COMPONENTS
Bob Bishop's Hires Saucer Invasion		TAPE	$ 15.00	COMPUTER COMPONENTS
Bingo for a Printer		TAPE	$ 10.00	Darrell's Appleware
Beat the House		TAPE	$ 14.95	G2 Program Library
Biorhythm		TAPE	$ 10.00	Programmer Software
Bomber		TAPE	$ 9.95	Softape Software
Battleship/3D Tic-Tac-Toe		TAPE	$ 12.00	U-ASKED 4 IT
Blackjack		TAPE	$ 10.00	George W. Lee
Bull & Bears		TAPE	$ 12.00	Softape Software
Battleship		TAPE	$ 7.50	Earl Keyser
Color Graphics Game		TAPE	$ 20.00	Programmer Software
Casino Royale		TAPE	$ 12.00	U ASKED 4 IT
Code Breaker, Secret Writing		TAPE	$ 11.95	U ASKED 4 IT
Christmas Carol	8k		$ 12.95	Micro Users
Car Race Hires Graphics		TAPE	$ 7.50	Darrell's Appleware
Clinic		TAPE	$ 14.95	G2 Program Library
Devils Dungeon		TAPE	$ 10.00	Rainbow Computing
Dollars & Sense		TAPE	$ 14.95	G2 Program Library
Dr. Apple–ELIZA		TAPE	$ 5.00	Earl Keyser
Drawing	8k	TAPE	$ 6.95	Innovative Computer
DOJOWO		TAPE	$ 10.00	Programmer Software
Desert/Arctic Survival		TAPE	$ 10.00	Earl Keyser
Don't Fall	8k	TAPE	$ 6.95	Innovative Computer
Dragon Maze, Digital Derby, Saucer War		TAPE	$ 2.00	*Softape Software
Electric Crayon	8k		$ 17.95	Micro Users
Escape	16k		$ 12.95	Micro Users
Fast Gammon		TAPE	$ 20.00	Quality Software
Foreign Legion Commando		TAPE	$ 9.95	U ASKED 4 IT
Funonyms		TAPE	$ 12.00	Speakeasy Software
Fly-Reaction Game		TAPE	$ 5.00	Earl Keyser

*Membership required to obtain these programs

ENTERTAINMENT

PROGRAM	MEMORY	MEDIA	PRICE	SOURCE
Graphics Package		TAPE	$ 3.00	ASTRO Graphics
Hangman/Concentration		TAPE	$ 12.00	U ASKED 4 IT
Horse Race		DISK	$ 25.00	Trans Data
Hires Life		TAPE	$ 10.00	Rainbow Computing
Hires Paddle Drawing Routine		TAPE	$ 20.00	Computer Components
Ham Radio (Send/Receive)		TAPE	$ 18.00	C H Galfo
Hot Air Balloon		TAPE	$ 10.00	Earl Keyser
Hyper-Life, & Graphic Demo		TAPE	$ 2.00	Softape Software
King		TAPE	$ 10.00	Programer Software
Keyboard Organ		TAPE	$ 18.00	Trans-Data
Keyboard Organ		DISK	$ 25.00	Trans-Data
Keyboard Organ	4k	TAPE	$ 6.95	Innovative Computer
Life		TAPE	$ 5.00	Earl Keyser
Life		TAPE	$ 6.50	ASTRO Graphics
Life (high speed, variable size)		TAPE	$ 12.00	C H Galfo
Maze Game	16k	TAPE	$ 12.95	Muse
Music Kaleidoscope		TAPE	$ 9.95	Softape Software
Microchess 2.0	16k	TAPE	$ 19.95	Personal Software
Microchess 2.0		TAPE	$ 15.00	Rainbow Computing
Modeler		TAPE	$ 10.00	Programmer Software
Music Box	8k	TAPE	$ 12.95	Micro Users
Music (3 Octaves)		TAPE	$ 20.00	Computer Components
Microtriva		TAPE	$ 12.00	Softape Software
Othello Game		TAPE	$ 10.00	Computer Components
Oregon Trail		TAPE	$ 10.00	Earl Keyser
Othello, Mastermind, Seven (card game)		TAPE	$ 2.000	*Softape Software
Pinball		TAPE	$ 10.00	Earl Keyser
Quiz Baseball	16k	TAPE	$ 7.95	Innovative Computer
Robots Game		TAPE	$ 7.50	ASTRO Graphics
Rocket Pilot, Saucer Invasion		TAPE	$ 12.95	Softape Software
Road Race, Space War		TAPE	$ 9.95	Softape Software
Save-a-Sketch	8k	TAPE	$ 6.95	Innovative Computer
Save-a-Story	8k	TAPE	$ 7.95	Innovative Computer
Star Wars, Space Maze		TAPE	$ 12.95	Softape Software
SIDE Shows	4k	TAPE	$ 12.95	Muse
Star Trek/Star Wars		TAPE	$ 10.00	Rainbow Computing
Stimulating Simulations— Applesoft II		TAPE	$ 14.95	Personal Software
Tank War	16k	TAPE	$ 12.95	Micro Users
Trap & Chase (with special hardware)		TAPE	$ 49.95	B & G Interfaces
Tic-Tac-Talker, Spectrum Analysis		TAPE	$ 19.95	Softape Software

*Membership required to obtain these programs

ENTERTAINMENT

PROGRAM	MEMORY	MEDIA	PRICE	SOURCE
U-Draw	16k	TAPE	$ 17.95	Micro-Users
UFO, Clean Sweep, Star Wars, Lunar Lander		TAPE	$ 2.00	*Softape Software
Warlords		TAPE	$ 12.00	Softape Software
10–4 Good Buddy		TAPE	$ 10.00	Earl Keyser
21		TAPE	$ 9.95	Softape Software

UTILITY & MISCELLANEOUS

PROGRAM	MEMORY	MEDIA	PRICE	SOURCE
Apple-Lis'ner		TAPE	$ 19.95	Softape Software
Apple Box Mini-Modem		TAPE	$ 18.50	Puget Sound Program Lib
Automatic Disk-to-Tape Back-up Copy		TAPE	$ 5.00	George W. Lee
Best of Bishop		DISK	$ 39.95	Softape Software
Common BASIC Programs (Applesoft)		DISK	$ 22.00	Basic Business
Circuit Logic Development Aid (255 gates)		TAPE	$ 10.00	Rainbow Computing
Common BASIC Programs		TAPE	$ 50.00	Programmer Software
Demos & Newsletter		TAPE	$ 6.45	Southeastern Software
Diet Planning		TAPE	$ 7.95	Ed Avelar
Drug Store's Patient/Drug File		TAPE	$ 50.00	Darrell's Appleware
Daily Calendar		TAPE	$ 50.00	Darrell's Appleware
Demo Cassette with Speechlab Model 20A			$189.00	Heuristics, Inc.
Data Save to Cassette		TAPE	$ 20.00	Computer Components
Disk-Compatible Author-Title Index Program		TAPE	$ 10.00	George W. Lee
HIRES Drawing Program		TAPE	$ 12.00	ASTRO Graphics
HIRES Graphics for F.P. Basic	24k	TAPE	$ 30.00	Computer Components
Introl Super-Voice Control/ graphics response		DISK		Mountain Hardware
Instant Library		DISK	$ 39.95	Softape Software
Instant Library		TAPE	$ 39.95	Softape Software
Library Pak 1		TAPE	$ 5.00	Puget Sound Program Lib
Library Pak 2		TAPE	$ 5.00	Puget Sound Program Lib
Library Pak 3		TAPE	$ 5.00	Puget Sound Program Lib
Library Pak 4		TAPE	$ 5.00	Puget Sound Program Lib
Menu		TAPE	$ 10.00	Earl Keyser
Memory Verify		TAPE	$ 5.00	Rainbow Computing
Memory Aide		TAPE	$ 18.00	Trans-Data
Memory Aide		DISK	$ 25.00	Trans-Data

*Membership required to obtain these programs

UTILITY & MISCELLANEOUS

PROGRAM	MEMORY	MEDIA	PRICE	SOURCE
No-RES Introl Demo-Voice Control of 6 Devices		TAPE/DISK		Mountain Hardware
Rainbow's Pot-of-Gold (49 BASIC Programs)		TAPE	$ 49.00	Rainbow Computing
Programmed Exercise Program			$ 19.95	Charles Mann
Super-Talker-Voice Control/ Response		TAPE/DISK		Mountain Hardware
Slow Scan TV Receiver (no hardware required)		TAPE	$ 18.00	C H Galfo
Text Pak #1 (including typesetting)		TAPE	$ 4.95	Ed Avelar
Trade Pak (6 for 6)		TAPE	Trade	Ed Avelar
Things To Do			$ 24.95	Charles Mann
Utility Sort		TAPE	$ 20.00	Darrell's Appleware
Utility Pak #1		TAPE	$ 5.95	Ed Avelar
Variable Message	8k	TAPE	$ 6.95	Innovative Computer
Word Processor	20k	TAPE	$ 50.00	Computer Components
Word Processor—Apple Editor		TAPE	$ 50.00	Darrell's Appleware
3-D Plot, Etch-a-Sketch, Star Burst		TAPE	$ 2.00	*Softape Software

*Membership required to obtain these programs

APPLE BIBLIOGRAPHY
Courtesy of: MICRO
MICRO IS PUBLISHED MONTHLY BY:
The COMPUTERIST, Inc
P.O. Box 3
Chelmsford, MA. 08124
$12.00 for 12 issues

Compiled by: William R. Dial

MOS Technology, Inc., 950 Rittenhouse Rd., Norristown, PA 19401 "6502 Programming Manual"

MOS Technology, Inc., 950 Rittenhouse Rd., Norristown, PA 19401 "6502 Hardware Manual"

Rankin, Roy and Wozniak, Steve, "Floating Point Routines for the 6502" Dr. Dobbs Journal 1, No. 7, pp. 17-19 (August 1976)

Baum, Allen and Wozniak, Stephen, "A 6502 Dissembler" Interface Age 1, No. 10, pp. 14-23 (September 1976)

Rankin, Roy and Wozniak, Stephen, "Floating Point Routines for 6502" Interface Age 1, No. 12, pp. 103-111 (November 1976)—See also DDJ 1, No. 7, pp. 17-19 (August 1976)

Apple Computer Inc., 20863 Stevens Creek Blvd., Cupertino, CA 95014, Byte *2*, No. 11, p 252 (Nov, 1977)

Ferruzzi, Arthur, "Inside the Apple II", MICRO, No. 1, pp 9-10 (Oct-Nov 1977)

Schwartz, Marc, "Ludwig von Apple II", MICRO, No. 2, p 19 (Dec 77-Jan 78).

Wozniak, Stephen, "Sweet 16: The 6502 Dream Machine", Byte *2*, No. 11, pp 150-159 (Nov. 1977)

Feagans, John, "A Slightly Sour SWEET 16", Byte *3*, No. 2, p 93 (Feb. 1978). Correction of a slight bug in the Wozniak article in Byte, Nov. 1977.

Electronics Warehouse Inc., 1603 Aviation Blvd., Redondo Beach, CA 90278, New Product Announcement.

Scogin, Tom, "AppleSOFT Benchmarks: Fast!", Kilobaud, No. 15, p 12 (Mar 78)

Carpenter, C.R., "Machine Language used in 'Ludwig von Apple II' ", MICRO, No. 3, p 8 (Feb - Mar. 1978)
Notes on an assembled version of the machine language used by Schwartz, MICRO, No. 2, p 19 in his music program.

Carpenter, C.R., "Printing with the Apple II", MICRO, No. 3, pp 13-16, (Feb-Mar, 1978)

Holt, Rod, "The Apple II Power Supply Revisited", MICRO, No. 3, p 28 (Feb-Mar. 1978)

Bishop, Robert J. "Star Wars" Kilobaud No. 14 pp 52-56 (Feb. 1978)

Helmers, Carl "An Apple to Byte", BYTE *3*, No. 3, p. 18-46 (Mar. 1978)

Anon., "Byte's Bits", BYTE *3*, No. 4, p 166 (April 1978)

Bishop, Robert J. "Rocket Pilot", Kilobaud No. 13, pg 90 (Jan. 1978)

Peoples Computers *6*, No. 6 (May/June 1978)
Cole, Phyllis "Apple II".

MICRO, Issue 4 (April/May 1978)
Carpenter, C.R. "Variables Chart".
Carpenter, C.R. "Apple II Printing Update".
Rowe, Mike "A Worm in the Apple".
Auricchio, Rick "An Apple II Programmer's Guide".

Creative Computing *4*, No. 4 (July/August 1978).
North, Steve "Apple II Computer".
Dawkins, Gary D. "High-Resolution Graphics for the Apple II".

MICRO, Issue 5 (June/July 1978)
Rowe, Mike "Half a Worm in the Apple". See also EDN May 20, 1978.
Suitor, Richard F. "Applayer Music Interpreter".
Carpenter, Chuck "Apple II Accessories and Software".

People's Computers *7* No. 2 (Sept/Oct., 1978)
Gaines John "Apple Math".

Conway, John "A Tape-to-Microcomputer-Hardware Interface Requires a Wealth of Microtechniques". EDN *23* No. 6 pg. 101-110 (March 20, 1978).

Hemenway, Jack E. "Add Floppies to Your Microcomputer to Form a Real Microcomputer System". EDN *23* No. 12 pg 98-107 (June 20, 1978)

Kilobaud Issue 23 (Oct., 1978)
Bishop, Robert J. "The Remarkable Apple II".

MICRO, No. 7 (Oct./Nov., 1978)
Auricchio, Rick "Breaker: An Apple II Debugging Aid".
Watson, Allen III "MOS 16K RAM for the Apple II.
Shryock, William H., Jr., "Improved Star Battle Sound Effects".
Schwartz, Marc, "Apple Calls and Hex-Decimal Conversion".
Eliason, Andrew H. "Apple II High Resolution Graphics Memory Organization".

Anon, "Tone Routine for Apple II". Southeastern Software Newsletter Issue No. 3 pg. 6 (Oct., 1978)

Haller, George, "Storing and Recovering Data in Applesoft II". Southeastern Software Newsletter Issue No. 2 pg. 4 (Sept, 1978)

Call - APPLE 1 No. 6
Williams, Don "Key Klicker Routine".
Anon, "Routine to Find Page Length".
Anon, "Printer Driver Fixes".
Anon, "Apple II Mini-Assembler".
Aldrich, Darrell "Use of Color Mask Byte in HIRES".
Anon, "Memory Map-Apple II with Applesoft Basic Loaded".
Anon, "List of Handy Calls".
Apple Computer Staff "System Monitor".
Huelsdonk, Bob "Memory Test".

Call - APPLE 1 No. 7 (August, 1978)
Golding, Val J. "A Disk Utility Program".
Backman, J.A. "Poor Man's HEX-DECIMAL-HEX Converter".
Thyng, Mike "Basic File Handling".
Apple Computer Staff "System Monitor".
Anon "Applesoft Zero Page Usage".
Huelsdonk, Bob "Routine to Print Free Bytes".
Huelsdonk, Bob "A Patch for Double Loops".
Apple Computer Staff "Loading Machine Language as Part of a Basic Program". Reprinted from Contact No. 1, May 1978.

Call - APPLE 1 No. 8 (Sept., 1978)
Aldrich, Ron "Convert".
Thyng, Mike "Arrays".
Chapman, Dan "Video Display Organization".
Anon "Routine to Save an Array". (reprinted from Apple Stems Vol. 1 No. 2 July, 1978)
Lam, S.H. "Monitor Commands from Basic".
Williams, Don "Linkage Routines for the Apple II Integer Basic Floating Point Package".
Hill, Alan G. "Return to TEXT from Graphics".
Anon "Integral Data IP 125-225 Driver".
Huelsdonk, Bob "Printer Driver Fixes".

Call - APPLE 1 No. 9 (Oct., 1978)
Cook, John B. "Applesoft Tone Routines".
Scott, Michael M. "A Brief History of Apple".
Anon, "Some Basic Entry Points".
Huelsdonk, Bob "Sample File Handler".
Golding, Val and Williams, Don "Apple II Integer Basic: Interpretation of Memory".
Golding, Val "Applesoft II Tokens".

apple computer inc.®
10260 Bandley Drive
Cupertino, California 95014
(408) 996-1010